NASA Contractor Report 2912

PCSYS: The Optimal Design Integration System Picture Drawing System With Hidden Line Algorithm Capability for Aerospace Vehicle Configurations

D. S. Hague and J. D. Vanderberg

CONTRACT NAS1-12977
DECEMBER 1977

NASA

NASA Contractor Report 2912

PCSYS: The Optimal Design Integration System Picture Drawing System With Hidden Line Algorithm Capability for Aerospace Vehicle Configurations

D. S. Hague and J. D. Vanderberg
Aerophysics Research Corporation
Bellevue, Washington

Prepared for
Langley Research Center
under Contract NAS1-12977

National Aeronautics
and Space Administration

Scientific and Technical
Information Office

1977

TABLE OF CONTENTS

TABLE OF CONTENTS (cont'd)

PREFACE

The ODIN procedure is a design analysis technique which allows the use of existing computer codes as part of a larger simulation. Communication of information among computer codes is accomplished by means of a data base repository accessible and managed by the ODIN executive computer code, ODINEX.

The objective of the contract was the development of improved independent geometry display program. The result is a system of computer codes for editing geometry monitoring geometric perturbations, and drawing realistic pictures of complex vehicle geometries.

NOTE ON MEASUREMENT SYSTEMS

Several vehicle drawings contained in this report include dimensional and performance data to illustrate features of the Aerophysics Research Corporation graphics codes. These numbers are quoted in U. S. Customary Units. These numbers may be converted to metric equivalent units using the following factors.

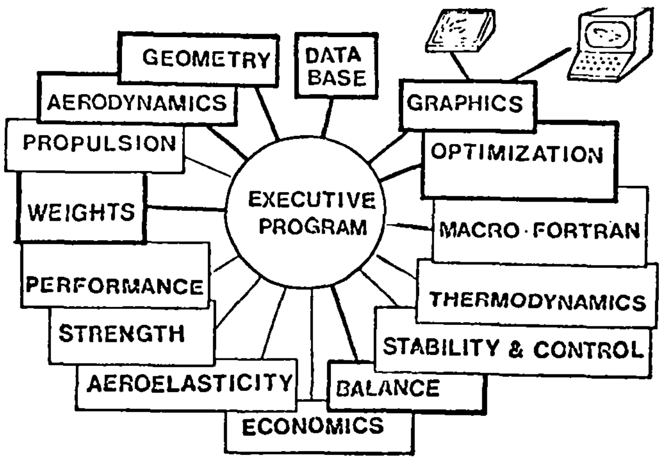
Quantity	U. S. Unit	Metric Unit	Conversion Factor
Length	feet	meter	0.30480
Area	square feet	square meter	0.092903
Volume	cubic foot	cubic meter	0.028317
Weight	pound	kilogram	0.453
Thrust force	pound	kilogram	0.453
Specific impulse	second	second	1.0
Velocity	feet/second	meters/second	0.30480

It should also be noted that certain of the computer graphics program input quantities which serve to position a drawing on the plot paper are set to nominal values in inch units. These values must be quoted in inch units when using the codes. For those interested in converting metric units to U. S. Customary Units, the multiplicative conversion factor for centimeters to inches is 0.39370.

SUMMARY

The PCSYS programs use a vehicle geometric definition based upon quadrilateral surface elements to produce realistic pictures of an aerospace vehicle. It is anticipated that the PCSYS system will be an important component of the Optimal Design Integration (ODIN) system, see sketch below. The PCSYS programs can be used to visually check geometric data input, monitor geometric perturbations, and to visualize the complex spatial inter-relationships between the internal and external vehicle components. The pictures constructed with PCSYS can be annotated with text information if desired.

PCSYS has two major component programs. The first program, IMAGE, draws a complex aerospace vehicle pictorial representation based on either an approximate but rapid hidden line algorithm or without any hidden line algorithm. The second program, HIDDEN, draws a vehicle representation using an accurate but time consuming hidden line algorithm. Program IMAGE is based on the widely distributed code of Gentry, ref. 1. HIDDEN is based on a code developed by researchers at Purdue University.



SKETCH OF THE ODIN SYSTEM FOR VEHICLE DESIGN SYNTHESIS

The contents of this report are arranged in three parts. Part I describes the original ODIN picture drawing program IMAGE. The second part describes the new picture drawing program HIDDEN as it has been used in the ODIN system. These adaptations consist of

a. An increase in the number of panels which may be employed in the definition of a vehicle surface

b. Conversion of the IBM 360 version of the program to the CDC 6600/7600 series computer

c. Extension of the code to configurations of the complexity presented in this report. Primarily this extension consisted of debugging the existing code.

Part III covers adaptation of the HIDDEN code to the ODIN system. Two programs have been written to accomplish this adaptation. These programs are CNVTHD and PLOTHD. Program CNVTHD automatically transforms the geometry input of the existing IMAGE code into the form accepted by HIDDEN. Program PLOTHD converts the output of HIDDEN into a form acceptable to the ODIN picture drawing programs. As a result of the present study effort, program HIDDEN has been completely integrated into the ODIN system.

PART I - PROGRAM IMAGE

The generation of geometry for aerospace configurations in digital format is one of the more tedious tasks in the design analysis process and is also difficult to check. The program described in Part I of this report provides a pictorial representation of the digital geometric input. This program capability provides a visual check for errors in geometry and a picture presentation which can be annotated with text information.

The IMAGE program is extracted from the program of reference 1. It has undergone extensive modification and simplification in an effort to provide a more useful addition to the ODIN (Optimal Design Integration) program library, references 2 and 3. Although developed primarily for use in the ODIN system, the IMAGE program is equally useful as an independent program. The program was written for the CDC 6600 computer. Some coding peculiar to the CDC machine must be changed for use on other machines but this coding is a small portion of the total program.

Versions of the IMAGE program are currently available for the CDC 6600 computer, reference 4, and the UNIVAC 1108, reference 5. The code is also operational on the CDC 7600 as a result of the present contract. Mr. Alan Wilhite of NASA's Langley Research Center has also prepared a version which operates on the "Prime" mini-computer.

SURFACE MODEL

The surface shape of the configuration is described to the IMAGE program by ordered sets of points in three dimensional space. The geometry is specified in 80 column (BCD) card format. It may be input directly or generated by another program and passed to the IMAGE program as a binary or BCD file. The program rotates the geometric data to prespecified viewing angles then transforms it into a plane coincident with the plane of the paper. The geometry is reordered into quadrilateral elements for plotting.

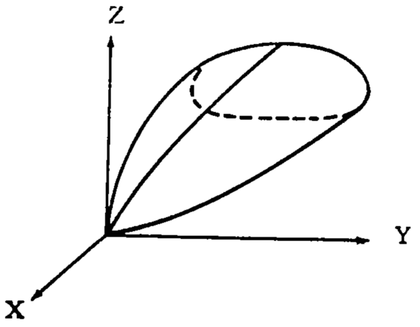
A grouping of four surface points is used to describe a quadrilateral surface element. An organization of a large number of related elements forms a component. A number of components may be used to give a complete description of the configuration. Each component is an independent unit of geometry which may be drawn separately or collectively with other components. In general, the equations for the geometry rotations described here apply to any orientation angle. The equations required to produce the perspective drawings are derived in the following paragraphs. Figure 1 is a collection of component geometries representing a shuttle orbiter configuration arranged in a three view drawing by the program.

Coordinate System

Each point on the surface is described by its coordinates in the body reference coordinate system.

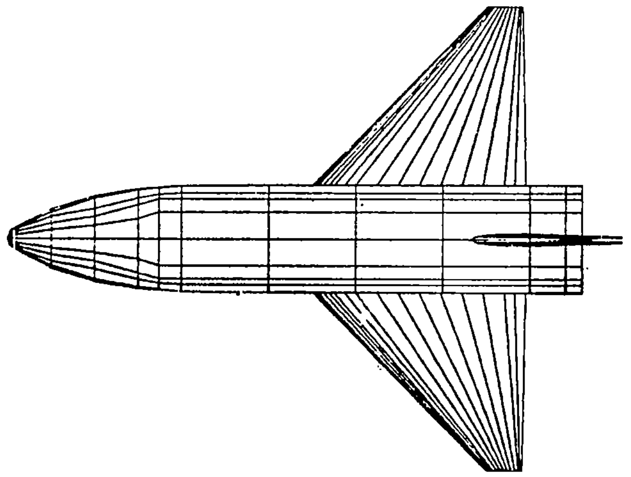
$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

The body reference coordinate system is assumed to be a conventional right-handed Cartesian system as illustrated below:

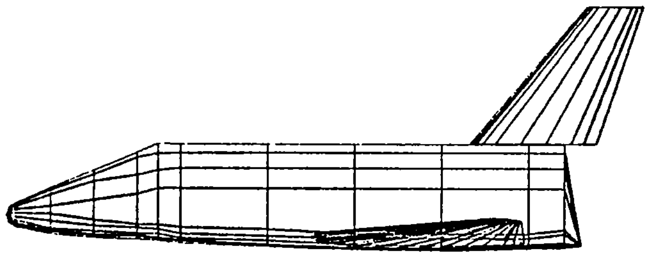


O D I N / I M A G E   PICTURE DRAWING PROGRAM

ORBITER DESIGN 25K P/L



```
VEHICLE CHARACTERISTICS

1.  MASS PROPERTIES

        LANDED WEIGHT                           196994.5
        ENTRY WEIGHT                            186710.7
        DESIGN CG. SUBSONIC.  25K P/L           69.95972
        DESIGN CG HYPERSONIC.  13K P/L          69.49021

2.  GEOMETRY

        BODY LENGTH                             110.0000
        TOTAL WING AREA                         2500.000
        CHORD. THEO ROOT                        50.69356
        CHORD. TIP                              6.651471
        ASPECT RATIO                            3.000000
        LEADING EDGE SWEEP ANGLE                45.00000
        TRAILING EDGE SWEEP ANGLE               0.000000
        ELEVON AREA. TOTAL                      375.0000
        EXPOSED WING LOCATION                   58.45000

3.  AERODYNAMIC CHARACTERISTICS

        DESIGN TRIM LIFT COEFFICIENT            .6257300
        DESIGN MINIMUM LANDING SPEED            187.8537
        MAX TRIM ALPHA HYPERSONIC DESIGN.COND   88.63326
```



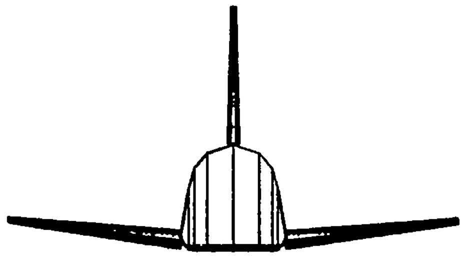
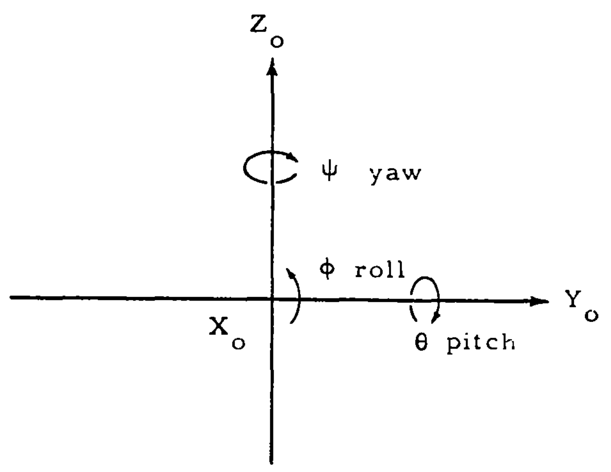


FIGURE 1   ILLUSTRATION OF PICTURE AND TEXT OPTIONS IN IMAGE.

Coordinate Transformations

To create the perspective drawings illustrated in this report each surface point on the body must be rotated to the desired viewing angle and then transformed into a coordinate system in the plane of the paper. With zero rotation angles the body coordinate system is coincident with the fixed system in the plane of the paper.



The rotations of the body and its coordinate system to give a desired viewing angle are specified by a yaw-pitch-roll sequence (Ψ,θ,ϕ). The rotation is given by the following relationship:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} \phi \end{bmatrix} \begin{bmatrix} \theta \end{bmatrix} \begin{bmatrix} \psi \end{bmatrix} \begin{bmatrix} X_o \\ Y_o \\ Z_o \end{bmatrix}$$

This sequence is important to remember when describing the desired viewing angles to the IMAGE program.

The rotation matrices ψ, θ and ϕ are given by:

$$[\psi] = \begin{bmatrix} \cos\psi & \sin\psi & o \\ -\sin\psi & \cos\psi & o \\ o & o & 1 \end{bmatrix}$$

$$[\theta] = \begin{bmatrix} \cos\theta & o & -\sin\theta \\ o & 1 & o \\ \sin\theta & o & \cos\theta \end{bmatrix}$$

$$[\phi] = \begin{bmatrix} 1 & o & o \\ o & \cos\phi & \sin\phi \\ o & -\sin\phi & \cos\phi \end{bmatrix}$$

or

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = [E] \begin{bmatrix} X_o \\ Y_o \\ Z_o \end{bmatrix}$$

where $$[E] = [\phi][\theta][\psi]$$

Since each point on the surface is given by its coordinates in the X, Y, Z system, its position in the fixed coordinate system (X_o, Y_o, Z_o) may be found by the inverse of the above process:

$$\begin{bmatrix} X_o \\ Y_o \\ Z_o \end{bmatrix} = [E]^{-1} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

If this operation is carried out, the resulting relationship is obtained.

$$\begin{bmatrix} X_o \\ Y_o \\ Z_o \end{bmatrix} = \begin{bmatrix} \cos\theta\cos\psi & -\sin\psi\cos\phi+\sin\theta\cos\psi\sin\phi & \sin\psi\sin\phi+\sin\theta\cos\psi\cos\phi \\ \cos\theta\sin\psi & \cos\psi\cos\phi+\sin\theta\sin\psi\sin\phi & -\cos\psi\sin\phi+\sin\theta\sin\psi\cos\phi \\ -\sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

$$X_o = X(\cos\theta\cos\psi) + Y(-\sin\psi\cos\phi+\sin\theta\cos\psi\sin\phi) + Z(\sin\psi\sin\phi+\sin\theta\cos\psi\cos\phi)$$

$$Y_o = X(\cos\theta\sin\psi) + Y(\cos\psi\cos\phi+\sin\theta\sin\psi\sin\phi) + Z(-\cos\psi\sin\phi+\sin\theta\sin\psi\cos\phi)$$

$$Z_o = X(-\sin\theta) + Y(\cos\theta\sin\phi) + Z(\cos\theta\cos\phi)$$

We may now use these last two equations to transform a given point on the body (X, Y, Z) with a specified set of rotation angles (ψ,θ,ϕ) into the plane of the paper (the Y, Z system). With the CALCOMP library subroutines it is a simple matter to plot these data and to connect the related points with straight lines.

The above relationships completely describe the transformation required to rotate every point on the vehicle to the desired angle, then into the plane of the paper. However, the resulting drawing is difficult to interpret because all hidden lines are drawn. Further, since only one half, or one quarter (for symmetrical vehicles) of the coordinate points are usually input, some additional calculations are desirable.

The surface points are grouped into quadrilateral surface elements for analysis by the program. Each quadrilateral is drawn as an individual 'curve.' The collection of all 'curves' represents the complete configuration. The technique also provides a convenient means of limiting the drawn lines to those normally seen by the viewer at the defined viewing angles.

Each input element is replaced by a plane quadrilateral surface made up of the four lines connecting the points. The quadrilateral characteristics are used to determine the visibility of the four lines. The quadrilateral characteristics include the area, centroid and the direction cosines of the surface unit normal. The surface unit normals may be transformed through the required rotation angles just as was done for the individual points. The resulting value of the component of the unit normal in the X_o direction (out of the plane of the paper) may be found from the following equation:

$$n_{x_o} = n_x(\cos\theta\cos\psi)+n_y(-\sin\psi\cos\phi+\sin\theta\cos\psi\sin\phi)+n_z(\sin\psi\sin\phi+\sin\theta\cos\psi\cos\phi)$$

where n_x, n_y, n_z are the components of the surface unit normal in the vehicle reference system.

If n_{x_o} is positive, then the surface element is facing the viewer. If n_{x_o} is negative, the element faces away from the plane of the paper. This result is used in the program to provide the option of deleting most of those elements on a vehicle that normally could not be seen by a viewer. The picture is thus made more realistic and easier to interpret. Confusing elements which are on the back side of a component do not appear. No criterion is provided, however, for the deletion of those elements that face the viewer but are blocked by other body components. If desired, more realistic drawings may be obtained by selective deletion of sections or half sections and by a proper selection of viewing angle. It should be noted that in certain open bodies portions of the outline of a body may be defined by quadrilateral elements which face away from the viewer. In such cases portions of the body outline will be deleted by the approximate hidden line algorithm contained contained in IMAGE. This problem can be avoided by use of the true hidden line algorithm described in Part II of the present report.

PROGRAM USAGE

The computer program usage requirements described in this section are generally oriented toward the CDC 6600 computer version and specifically toward the Langley Research Center (LRC) installation. The actual program input requirements described are largely applicable wherever the program is installed but the control cards for the retrieval and execution of the program will differ from computer to computer and from installation to installation.

The use of the program requires three types of input; the control cards, the actual program input and the post processing instructions for generating CALCOMP plots. Figure 2 illustrates the deck setup for executing the program and indicates the separate submittal of a plot request card. One input case is illustrated. Multiple cases may be computed by repeating the case data illustrated.

Control Cards

The program input data is preceded by the operating system control cards required to retrieve from permanent storage and execute the program. Figure 3 illustrates the control cards for three different ways of using IMAGE at LRC. Figure 3A shows the method of execution from the stored machine language program. Figure 3B illustrates a compile, load and execute sequence from stored source code, assuming program modifications are desired. Figure 3C illustrates the use of IMAGE within the ODIN (Optimal Design Integration) system. See reference 2 for ODIN system usage.

The dashes on the JOB, USER and REQUEST cards indicate missing information which the user must supply in accordance with LRC computer complex accounting procedures. Further, the wedge number (program storage location in data cell) is subject to change. The latest wedge number is available from LRC.

The user must supply the estimated run time in CPU seconds, the octal field length for the job and the number of operating systems (O/S) calls. These are given in the above order on the JOB card. Typical values for a single case are tabulated below:

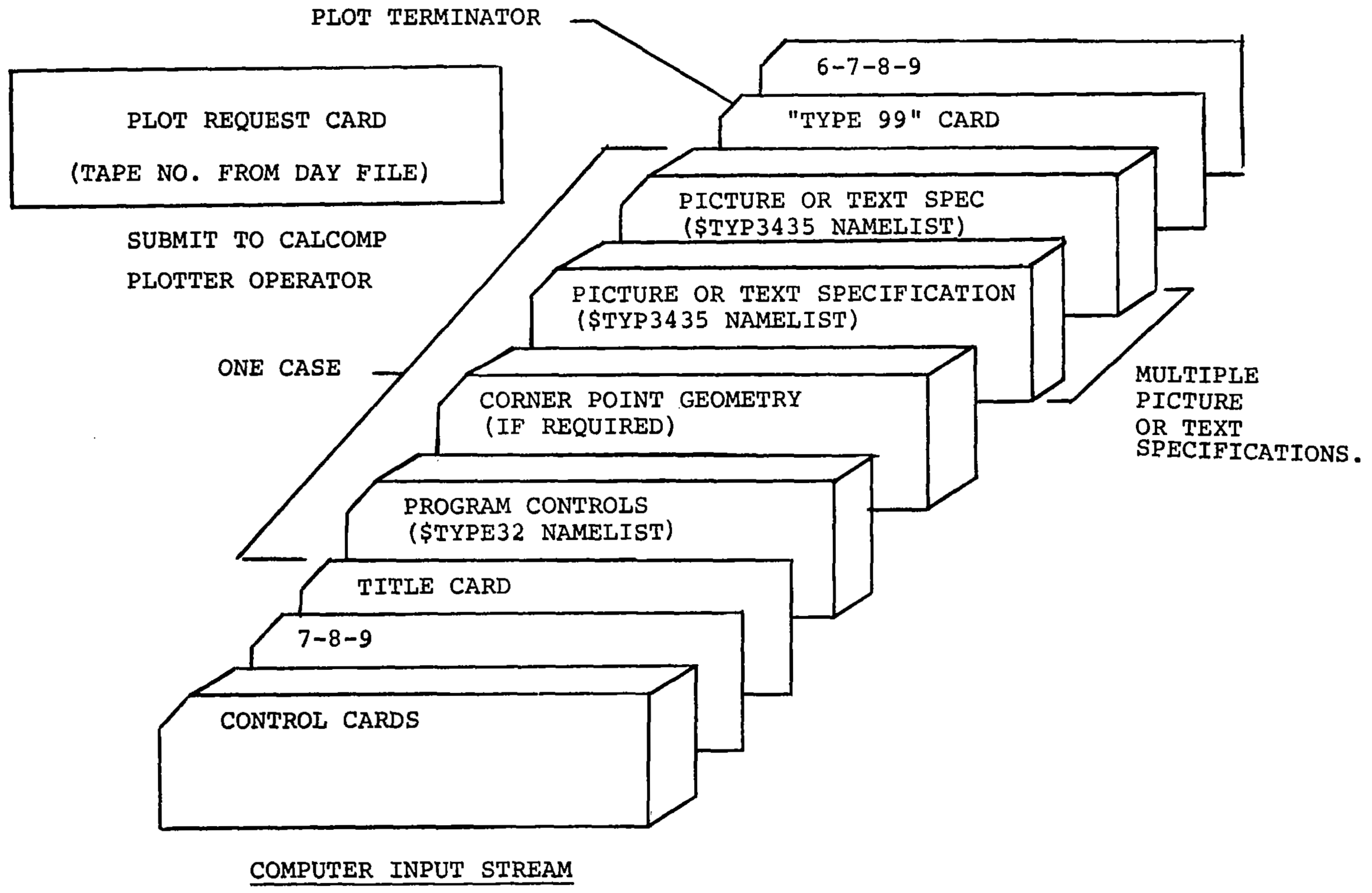


FIGURE 2 DECK SETUP FOR IMAGE.

```
RUNID,T20,CM20000.
ACCOUNT CARD
CHARGE CARD
GET,IMAGE/UN=727850N.
IMAGE.
PLOT.
7-8-9
(IMAGE CASE DATA)
6-7-8-9
```

FIGURE 3A EXECUTION OF STORED PROGRAM

```
RUNID,T20,CM45000.
ACCOUNT CARD
CHARGE CARD
FETCH,A3647,SOURCE.
RUN,S,,,SCFILE.
LGO.
PLOT.
7-8-9
(MODS TO SOURCE PROGRAM,IF ANY)
7-8-9
(IMAGE CASE DATA)
6-7-8-9
```

FIGURE 3B COMPILE, LOAD, AND EXECUTE

```
'EXECUTE IMAGE'
(IMAGE CASE DATA)            } REQUIRED ONLY IF PLOTS
7-8-9                        } ARE REQUESTED
'EXECUTE PLOTSV'
7-8-9
```

FIGURE 3C EXECUTE IMAGE WITHIN ODIN SIMULATION

FIGURE 3. ILLUSTRATIONS OF CONTROL CARDS REQUIRED FOR PROGRAM IMAGE

	CPU	FIELD LENGTH
EXECUTE ABSOLUTE BINARY PROGRAM	20	35000
COMPILE, LOAD AND EXECUTE	40	45000
EXECUTE WITHIN THE ODIN SYSTEM	30	56000*

*Minimum size for the executive system (reference 2).

The tabulated values above will vary with the complexity of the configuration (number of elements), the number of cases and the number of plots requested. A good rule to follow is to allow ample CPU and O/S calls in the first run, then use the day file results to estimate these parameters for similar configurations.

General Program Input

The IMAGE program input will typically consist of four types of input.

1. Title card in free field format.

2. Program controls ($TYPE32) in NAMELIST format.

3. Surface model data as formatted corner-point geometry.

4. Picture specifications ($TYP3435) in NAMELIST format.

The program controls and picture specifications are input in standard NAMELIST input format. Two NAMELIST reads are provided for this purpose as illustrated in figure 4.

The use of NAMELIST input was chosen for the following reasons:

1. It is a simple name oriented input easily understood by most computer users.

2. The format is standard and does not require relearning from program to program.

3. It is easily modified by the engineer or programmer when adding input variables to the program.

```
{TITLE (one card - first 59 characters)

( $TYPE32
{      [Geometry Scaling and Control Options]
( $

(      [Element Data - only if unit 5 ]
{      [was specified by the data set ]
(      [above                         ]

( $TYP3435
{      [Picture Drawing or Text Options]
( $

   $TYP3435
(      [Picture Drawing or Text Options]
{      [ (if                           ]
(      [LAST=1                         ]
   $

{END OF IMAGE DATA              col. 71 ———→99
```

FIGURE 4   ILLUSTRATION OF INPUT STREAM TO IMAGE FOR TWO VIEWS.

When a NAMELIST read is encountered in the program, the entire input file is scanned up to an end-of-file or a record with a dollar ($) in column 2 followed immediately by the NAMELIST name requested by the programs. Succeeding data items are read until a second dollar ($) is encountered signifying the end of the NAMELIST. Any data on the input file before the requested NAMELIST is found will be ignored. All data between the opening and closing dollar is interpreted by the NAMELIST input routine. The data item within the NAMELIST statement may be in any of three forms:

$V = C,$

$A = D_1,\ldots,D_j,$

$A(n) = D_1,\ldots,D_m,$

V is a variable name; C is a constant; A is an array name and n is an integer constant subscript, $D_1,\ldots D_m$, are simple constants or repeated constants of the form k*C, where k is the repetition factor. Constants may be real, integer, hollerith or logical. Hollerith constants are preceded by nH where n is the number of characters in the hollerith constant. Logical constants are of the form .TRUE. (or T) or .FALSE. (or F).

Data items and constants must be separated by commas. The number of constants, including repetitions given for an unscripted array must equal the number of elements in that array. For a subscripted array name, the number of constants need not be equal but may not exceed the number of array elements needed to fill the array. More than one card may be used for input data and arrays may be split between cards. All except the last record must end with a constant followed by a comma and no sequence numbers may appear. The first column of each record is ignored. The set of data items may consist of any subset of the variable names associated with the NAMELIST name and the name need not be any particular order. More details on the use of NAMELIST are available in any FORTRAN user's guide, but the above description should be sufficient for the operation of the IMAGE program.

The first list ($TYPE32) is used for specifying general data related to orientation, translation and scaling of the geometry. It also provides for setting flags pertaining to printing and geometry file control.

The $TYPE32 data set is followed by element geometry data if (described below) the INPUT file is specified in the above data set. Alternately the data may be read from the internal unit number 8. The geometry data is read, translated and scaled according to user instructions, then placed on a scratch unit (TAPE3) in binary format for use in the remainder of the program.

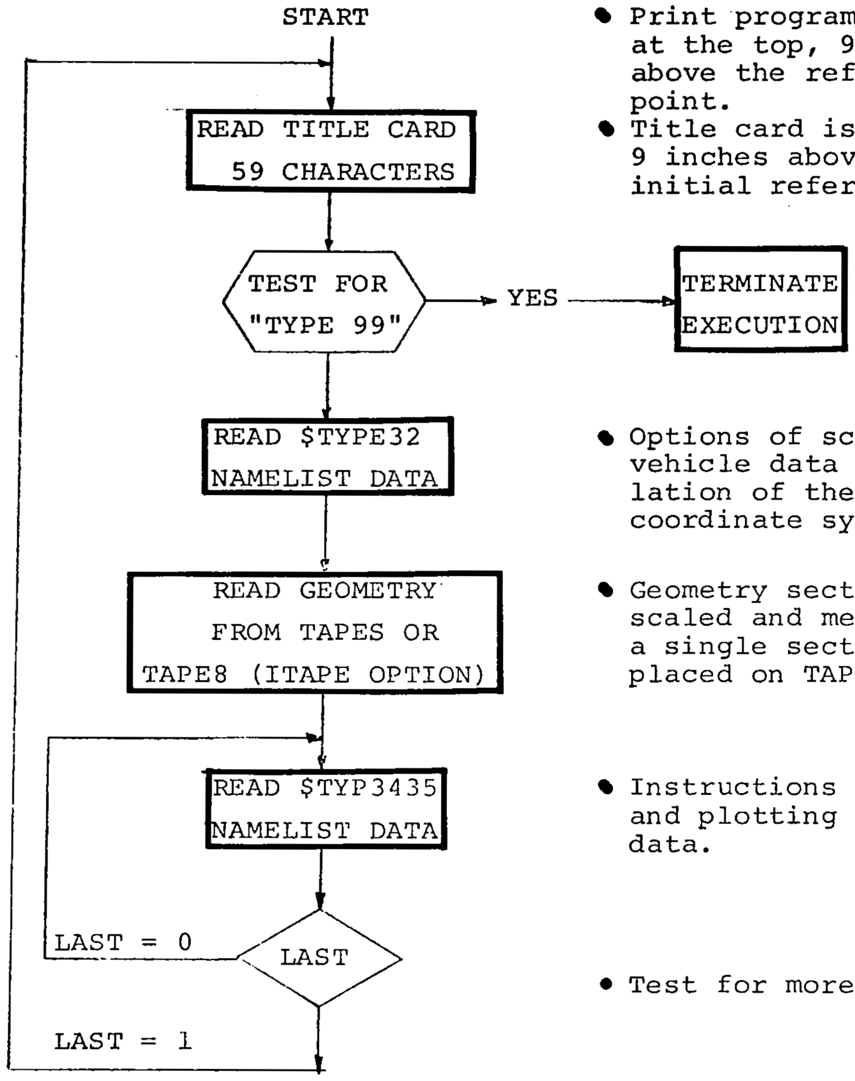
After geometry data (if any) the second NAMELIST ($TYPE3435) is input. This data set provides input for picture control options, etc., required to generate the desired pictures. Figure 4 illustrates an input stream to IMAGE. Any number of views may be specified by repeating the $TYP3435 data set. The last data set should specify LAST = 1.

The parameter, LAST, terminates the picture sequence for the current geometry which was temporarily placed on TAPE 3. The program logic returns to read a new TITLE card. The input flow is illustrated in figure 5. If a TITLE card is present in the input stream, additional geometry will be read from TAPE5 or TAPE8 and placed on the temporary storage file, TAPE3. A new sequence of pictures will be expected after the geometry is read. The execution is terminated by placement of a special (type 99) card in the input stream in place of a TITLE card. The special card must contain the integer, 99 in columns 71 and 72. The following paragraphs describe each input type in detail.

Title Card. - The TITLE card must be the first card in the input sequence. This card may contain from 1 to 59 characters (columns) of information which will be printed at the top of the frame, 9-inches above and 3-inches to the right of the initial reference point. A TITLE card must be present for each set of geometry which is to be processed by the program. Failure to supply this card will result in an input error. If no title is desired, a blank card must be inserted. The characters (99) in columns 71 and 72 of the TITLE card will cause normal termination of the program.

Program Controls. - The $TYPE32 data set is a NAMELIST input set consisting of input instructions for the geometry data, scaling, translation and orientation of the data, and printing instructions for the quadrilateral element characteristics. Figure 6 summarized the NAMELIST names and descriptions for the input set.

This section discusses each NAMELIST input in detail. Each paragraph is headed by the name and default value for the



- Print program header at the top, 9.5 inches above the reference point.
- Title card is printed 9 inches above the initial reference point.
- Options of scaling the vehicle data and translation of the vehicle coordinate system.
- Geometry sections are scaled and merged into a single section and placed on TAPE3.
- Instructions for scaling and plotting the picture data.
- Test for more geometry.

FIGURE 5 ILLUSTRATION OF INPUT FLOW LOGIC FOR IMAGE.

NAMELIST NAME	DEFAULT VALUE	DESCRIPTION
DELX	1.0	X-translation of scaled input data.
DELY	1.0	Y-translation of scaled input data.
DELZ	1.0	Z-translation of scaled input data.
IORIEN	0	Integer definition of a element orientation. = 0 Cross section input mode. = 1 Streamwise input mode. = 2 or 3 See text.
IREW8	0	Control integer for logical unit 8. file position. = 0 Rewind tape 8 before reading. = 1 Do not rewind 8 before reading.
ISTAT3	1	Number of vehicle components.
ITAPE	0	Geometry control integer. = 0 Geometry from unit 5. = 1 Geometry from unit 8 (coded). = 2 Geometry from unit 8 (binary).
PRINTS	0	Print flag. = 0 No printing. = 1 Print quadrilateral data.
XSC	0.0	X-scale factor.
YSC	0.0	Y-scale factor.
ZSC	0.0	Z-scale factor.

FIGURE 6 $TYPE32 NAMELIST INPUT SET.

variable listed. The heading is followed by a description of the input variable. If the default value is acceptable, the user need not define a value for it in the $TYPE32 NAMELIST data set.

PRINTS = 0	PRINTS is an integer variable which may have meaningful values of zero (0) or one (1). The variable controls the printing of detailed quadrilateral element characteristics. 0 Element characteristics will not be printed. 1 Element characteristics will be printed.
IORIEN = 0	IORIEN is an integer variable which defines the element orientation for the data. All vehicle components must have the same orientation. 0 Normal mode using cross sections. 1 Geometry is input in streamwise strips. 2 Geometry is input in streamwise strips. For each strip of elements, the first coordinate point in the right-hand strip of points is not used in the formation of the leading edge element but is ignored by the program. 3 Same as = 2 except the left-hand point is ignored in the formation of the leading edge elements.

Usually the data is described with IORIEN values of 0 or 1. Values of 2 or 3 will give correct pictures when the data-point slip methods of reference 1 are used to input the geometry data. Vehicle components with different values for IORIEN cannot be drawn correctly on a single picture frame.

The scale factors and associated translation increments described below are generally used to transform the configuration geometry to a more convenient form for plotting. The factors are frequently used to move the vehicle reference axis so that the vehicle center corresponds approximately to the coordinate system origin. For a vehicle with its nose at X = 0.0 this is accomplished by using a DELX value (see below) of about one-half of the vehicle length. This simplifies the selection of picture scales to keep the vehicle within the imaginary picture frame.

The original geometry data on tape 5 or tape 8 is not changed by the use of the scale factors. They are applied to the geometry stored on the temporary file TAPE3. The data is transformed using the following equations for X, Y and Z.

$$X_{new} = X_{input} \cdot (XSC) + DELX$$

$$Y_{new} = Y_{input} \cdot (YSC) + DELY$$

$$Z_{new} = Z_{input} \cdot (ZSC) + DELZ$$

The above scaling terms are defined as follows:

XSC	= 1.0	Scale factor to be multiplied by X_{input}.
YSC	= 1.0	Scale factor to be multiplied by Y_{input}.
ZSC	= 1.0	Scale factor to be multiplied by Z_{input}.
DELX	= 0.0	X increment to be added to $X_{input} \cdot (XSC)$.
DELY	= 0.0	Y increment to be added to $Y_{input} \cdot (YSC)$.
DELZ	= 0.0	Z increment to be added to $Z_{input} \cdot (ZSC)$.

Usually the configuration geometry consists of more than one component. In the data format of the IMAGE program, the data for each component is terminated with an integer flag referred to as a status flag as described under Surface Model Data (below). The merging of the geometric components is desirable for plotting purposes. Therefore, the geometric components are merged into a single component as the geometric data is being transferred to the temporary file, TAPE3. The user of the program must specify the number of components to be merged for each picture sequence. This specification is accomplished by the integer input variable ISTAT3.

ISTAT3 = 1	Integer variable number of vehicle components with Status = 3 in the vehicle geometry which the user wishes to plot as a unit. The program will count the number of Status = 3 in the geometry deck and when the count reaches this input value, the program will proceed to the plot options for the vehicle components which have just been read.

Several options are available to the user of the program for accessing geometric data. Alternate files may be employed and alternate formats may be specified. The input parameter for controlling the above options is the integer variable ITAPE.

ITAPE = 0 Geometry tape control integer variables with the following possible values:

= 0 Geometry data (type 3) will be read from Tape 5 (geometry data cards are loaded along with picture-data control cards).

= 1 Geometry data (Type 3 will be read from the geometry storage tape (Tape 8) in coded format.

= 2 Geometry data (Type 3) will be read from the geometry storage tape (Tape 8) in binary format.

The IMAGE program provides a flexible means of controlling the alternate geometry file, TAPE8. Multiple components may be stored on TAPE8. These components may be extracted in sequential groups or plotted individually. The input parameter for rewinding the geometry tape (TAPE8) is IREW8. Usually the file is rewound for the first sequence of pictures, then through the use of the input variable ISTAT3, additional groups of components can be extracted for plotting purposes.

IREW8 = 0 Integer variable to control the position of Tape 8 just before the geometry data are read from it.

= 0 Rewind Tape 8 and then read geometry data from it.

= 1 Do not rewind Tape 8, but start reading geometry data from it in its current position.

The integer, IREW8, permits the user to store more than one group of vehicle geometrics on the geometry tape, then collect and plot them by groups.

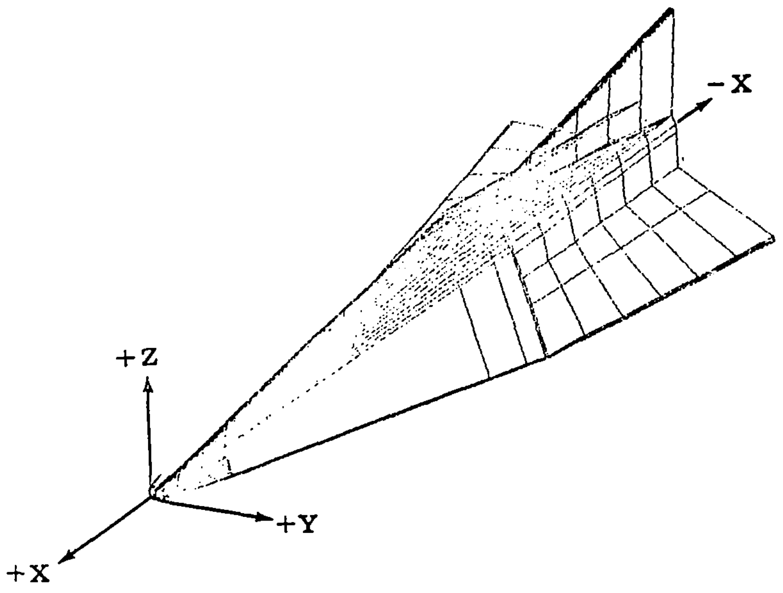
Surface Model Data. - The program accepts the geometry in element data form only. No simplified geometry techniques are employed. Several auxiliary programs are available which convert simplified geometry to element data format suitable for input to the IMAGE program. Reference 1 is an example of such a program. It generates element data using several mathematical surface generation options and includes an aircraft geometry option which provides a convenient means of generating aerodynamic surfaces.

The element data in IMAGE may be read from input or from an alternate unit. If the INPUT file is selected, the actual data cards are merged with the other input data as illustrated in figures 2 and 4.

If the alternate input unit (logical unit 8) is selected for element data input, the data may be read in coded (same as INPUT cards) or binary (fast read) mode. In order to use the binary mode, the data must have been written in binary mode. Reference 1 generates element data in coded format. The binary mode is very useful for improving efficiency particularily when the geometry must be regenerated many times for a problem solution. Regeneration of geometry is often desirable when studying the effects of geometric shapes. Regeneration will be required when optimizing geometric shape in an ODIN simulation.

Element Data Format. - The element data method uses a large number of surface coordinate points on the surface of the configuration. The points can be ordered around a station contour or in a streamwise manner. Each point consists of an X, Y, Z coordinate set and a status flag. Each card contains two points.

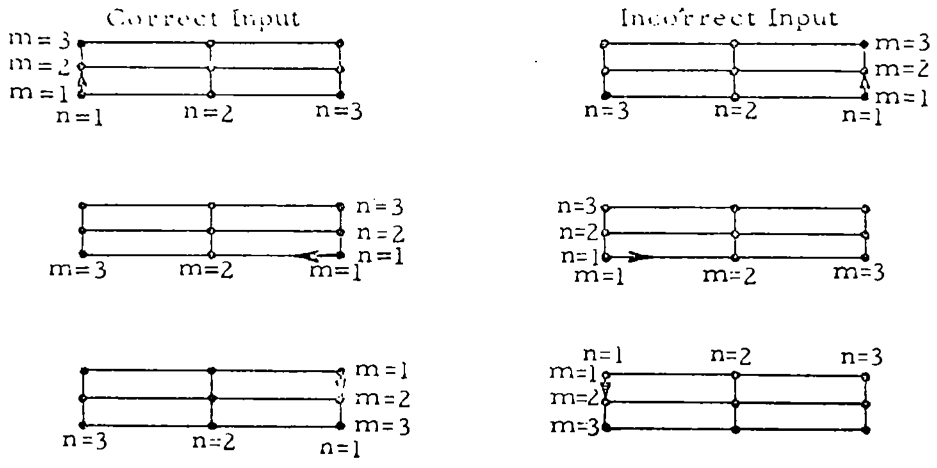
The coordinate system used for all the geometry data is shown in the figure below. For symmetrical vehicles it is standard practice to input the left side of the vehicle only. There is only one input method for the corner-point geometry and since any of the auxiliary geometry programs finally produce geometry data in surface-element form, it is important that the methods and nomenclature used with this method be clearly understood. It is, therefore, recommended that the input instructions for the surface-element method be studied before an attempt is made to use the method or write a geometry generation routine.



The geometric input data in this method include the coordinates of a large number of points on the vehicle surface. The input data are organized in a manner that permits the description of a vehicle on a component buildup basis. This gives increased flexibility in shape description and makes it possible to draw exploded views by physically separating the components of the vehicle. Because of possible changes in the surface contours of a configuration, or replacement of the entire component, it may also be desirable to divide the configuration into several components. This permits easy changes either in a manual or automated mode such as ODIN. Each component of a configuration is further divided into a number of sections each defined by a group of points in space. In practice, the surface coordinates are usually recorded from cross-section drawings of the vehicle in such a way that each point need be read only once (even though it may be a member of as many as four adjacent quadrilateral elements). Each point is defined by its three coordinates and a status flag that indicates whether it is the first point of a new section, a continuation of a group of points defining a section, the beginning of a new section, or the last point of the component. The program uses the status flags to determine how the input points are to be related to form the quadrilateral elements, and how the elements are combined to form a component.

The first question that the user asks when starting to load the element geometry is the order in which the surface points are entered. The basic rules to be followed are given below. The rules discussion will be followed by a discussion of a visual technique that many users will find helpful in determining the proper loading order.

For the purpose of organizing the input data for computation, each point is assigned a pair of integers, m and n. These integers are not actually input to the program (they are calculated internally) but their use in the following discussion will provide a better understanding of the input data organization. For each point, n identifies the "column" of points to which it belongs, and m identifies its position in the "column," i.e., the "row." The first point of a "column" always has m = 1. To insure that the program will compute outward normal vectors, the following condition for the order to input points must be satisfied. If an observer is located in the outside the component and is oriented so that locally he sees points on the surface with m values increasing upward, he must also see n values increasing toward the right. Strict adherence to this simple rule will always lead to a correct set of input geometry data. Examples of correct and incorrect input are shown in the sketches below. In these pictures the exterior of the configuration lies above the paper, and the interior of the configuration lies below the paper. The arrows indicate the order of reading the points.



Associated with each input point is an input quantity called its status. The first point of each new section has Status = 2. Except for the first n-line of a component, the first point of each n-line has Status 1. The last point of the component of the vehicle has Status 3. All other points have

Status = 0 (i.e., they may be left blank on the input sheet). The IMAGE program plots the picture according to components ending with a Status = 3.

The simple visual technique described below is helpful in determining the proper order of the input points:

1. First, assume that you are holding in your hand a small model of the vehicle shape. Many program users find it helpful to construct a small paper model to help in visualizing the geometry loading procedure. On this model draw lines to represent the elements to be loaded for a given vehicle section.

2. Next, decide which strips of elements are to constitute "columns" and which "rows." In most problems one of two procedures is selected - either a "column" of elements starts at the bottom of the shape and continues around to the top, roughly following vehicle cross-section lines, or a "column" is oriented so that it starts at the front part of the vehicle and runs aft toward the rear.

3. Hold the model out in front of you and rotate it until the columns are vertical with the first row of elements at the bottom. This procedure should be used regardless of what part of the vehicle is being loaded - the body, fin, inside of fin, etc. Always orientate the model so that you are looking at the section to be loaded (from the outside, looking at the surface) with the columns running vertical, and the rows running horizontal.

4. Now that you have the section being loaded oriented in front of you, with the columns vertical, apply the following cardinal geometry rule:

   > If a column of data points are loaded from the bottom to the top, then the next column of points (starting with a Status = 1) must be to the right.

All of the geometry input data for this geometry option are input on "Type 3" element data cards (an integer 3 in column 72). Each card contains the X, Y, Z coordinates and status flag for two points on the body surface. Every card in the element geometry deck must contain two surface points except the last card, which may have only the first surface point coordinates and status filled in. If a particular line of vehicle points is odd in number, then it is usually advisable to repeat the last point (a dummy point) so that the last card will have two sets of point data. This permits the shifting of configuration components without disrupting other components. A description of the input card for element data is shown in figure 7.

Picture Specifications. - The picture control data set is a NAMELIST input set called $TYP3435 which is summarized in figure 8. It controls the drawing of pictures and the printing of text. A typical deck setup will consist of several data sets of this type, one for each picture desired. The program will always try to read one data set. Additional views or text information may be generated or added by the inclusion of additional $TYP3435 data sets. The input integer controlling this function is LAST.

LAST = 0	Integer variable controlling the program flow logic after drawing a picture or printing text.
	= 0 Return for $TYP3435 data set.
	= 1 Return for new TITLE card (may result in program termination or the reading of more geometric data).

If the variable LAST is set to one (1), the reference axes system of the plotting device is moved to a new frame position specified by the user. The input variables controlling the axis translation are XMOVE and YMOVE.

XMOVE = 17.0	The translation of the reference axis system in the X-direction (in inches) following the last $TYP3435 data set.
YMOVE = 0.0	The translation of the reference axis system in the Y-direction (in inches) following the last $TYP3435 data set.

The values of XMOVE and YMOVE are made with respect to the original coordinate system reference specified upon execution of the program. The picture positioning parameters DXG and DYG described below have no affect on the XMOVE, YMOVE translation.

The program permits the user to specify the viewing angles with the three-axis system discussed in the early part of this report. The righthand rule is used for identifying the positive rotation angle of the vehicle geometry. The sequence of rotation is yaw, pitch and roll.

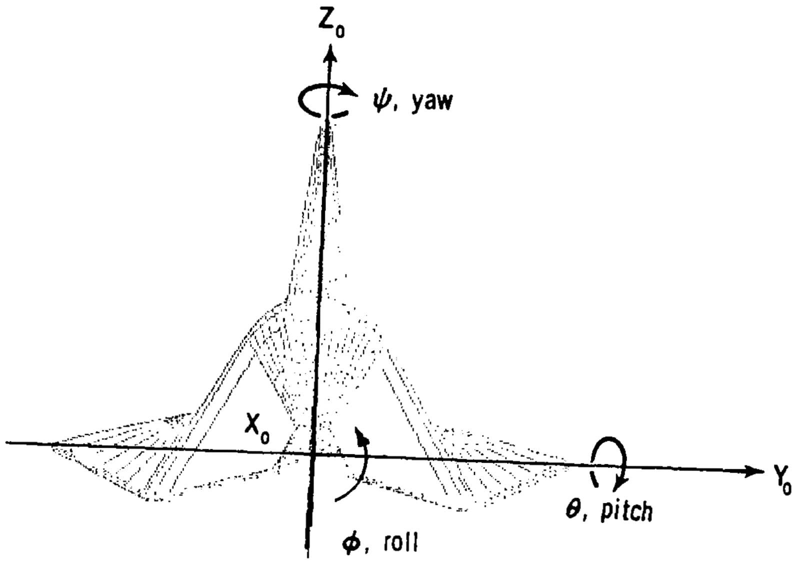
Column	Code	Explanation
1-10	X	X-coordinate of surface point (the value of X is written anywhere in this space with a decimal point and sign; usually input only if it is negative).
11-20	Y	Y-coordinate of surface point.
21-30	Z	Z-coordinate of surface point.
31	STAT	Status flag for the above set of coordinates (=2, 1, 0, or 3).
32-41	XX	X-coordinate of surface point.
42-51	YY	Y-coordinate of surface point.
52-61	ZZ	Z-coordinate of surface point.
62	STATT	Status flag for the above set of coordinates (=2, 1, 0, or 3).
66-68	CASE	Case number (right-justified integer)
69-70	SECT	Numbers or letters to identify the vehicle section. These must be legal machine characters.
72	TYPE	Card type number = 3.

FIGURE 7 ELEMENT DATA INPUT CARDS

NAMELIST NAME	DEFAULT VALUE	DESCRIPTION
DXG	0.0	Repositioning of X- and Y- reference point before plotting current picture - inches.
DYG	0.0	
HTEXT	0.14	Character height for text material - in.
IAREA	0.	Print control integer. = 0 No print. = 1 Print area of each section.
ICS	0.	Connectivity flag for quadrilaterals. = 0 Connect all four points. = 1 Connect points 1-2 and 3-4. = 2 Connect points 1-4 and 2-3. = 4 Do not connect points.
IQUAD	0.	Controls points to be drawn. = 0 Draw input points. = 1 Draw computed points on quadrilateral.
IREFL	1.	Reflected element flag. = 0 Draw input elements only. = 1 Draw reflected elements. = 2 Draw reflected elements (only one quadrant is input).
ISHAD	0.	Hidden line option. = 0 Do not plot hidden lines. = 1 Plot all lines.
LAST	0.	Program control flag. = 0 Return for $TYP3435 data set. = 1 Return for TITLE card (or Type 99 to terminate).

FIGURE 8A $TYP3435 NAMELIST INPUT SET.

NAMELIST NAME	DEFAULT VALUE	DESCRIPTION
PHI	0.	Roll angle (see below) - degrees.
PSI	0.	Yaw angle (see below) - degrees.
THETA	0.	Pitch angle (see below) - degrees.



SCAL	6.0	Frame size in inches. Geometry will be scaled to this dimension .
TEXT	F	Logical control variable for text input. = .TRUE. Text information will follow $TYP3435. = .FALSE. No text information will be input.

FIGURE 8B $TYP3435 NAMELIST INPUT SET. (CONTINUED)

NAMELIST NAME	DEFAULT VALUE	DESCRIPTION
XLG	*	Extreme forward X-dimension on input geometry - input units.
XRG	*	Extreme aft X-dimension on input geometry - input units.
YBG	*	Extreme bottom Y-dimension on input geometry - input units.
YTG	*	Extreme top Y-dimension on input geometry - input units.
XMOVE	17.0	Translation of X-reference after one complete case - inches.
YMOVE	0.0	Translation of Y-reference after one complete case - inches.

*Computed before. This input set is read but may be overridden (see text).

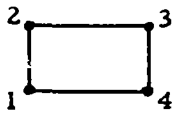
FIGURE 8C $TYP3435 NAMELIST INPUT SET (CONTINUED).

PSI = 0.	Yaw angle in degrees measured with respect to the positive Z-axis of the reference geometry coordinate system.
THETA = 0.	Pitch angle in degrees measured with respect to the positive Y-axis of the reference geometry coordinate system.
PHI = 0.	Roll angle in degrees measured with respect to the positive X-axis of the reference geometry coordinate system. Note that the positive X-axis is usually toward the nose of the vehicle.

The technique used for plotting geometric data is to treat each quadrilateral element as a five-point 'curve' in the image plane. The IMAGE program provides the user with the option of specifying the points on each quadrilateral element which will be connected when plotting the elements. The integer variable which controls this option is ICS.

ICS = 0 Integer variable controlling the points to be connected.

= 0	Connect all 4-points of each element.
= 1	Connect points 1-2 and 3-4 (see diagram below)
= 2	Connect points 1-4 and 2-3.
= 3	Do not connect points with lines



Quadrilateral Element

Usually the geometric data plotted by the IMAGE program is symmetrical about the X-Z plane and data is generally provided for only one side of the configuration. The program provides the user the option of plotting only the input geometry, the reflected geometry or alternately plotting both the input geometry and the geometric reflection of the input geometry. The input integer controlling this option is IREFL.

IREFL = 1 Integer variable controlling the plotting of reflection-elements as follows:

	= 0	Do not plot elements reflected to negative side of Y-axis.
	= 1	Plot reflected elements.
	= 2	Plot reflected elements (only one quadrant is input).

The plotting of all quadrilateral elements can often be distracting if not misleading with respect to appearance of the vehicle which the geometric data represents. The distraction is usually caused by the plotting of elements which face away from the viewer. The IMAGE program provides the user with the option of eliminating the above class of elements for the individual geometry components. The integer variable which controls this option is ISHAD. IMAGE does not provide any capability for omitting elements which face the viewer but are hidden by another component of the vehicle.

ISHAD = 0	Integer variable controlling the plotting of elements facing away from the viewer.	
	= 0	Do not plot elements that face away from the viewer (shadow elements).
	= 1	Plot shadow elements (elements facing away from viewer).

The IMAGE program provides the user the option of printing the surface area characteristics of the vehicle components as follows:

IAREA = 0	Integer variable controlling the printing of section areas.	
	= 0	Do not print section areas.
	= 1	Print out the area of each section.

Usually, the desired picture is represented by the actual corner points of the geometric data. However, for some types of analysis, it is desirable to view the corner points of the actual quadrilateral elements since the latter represents the data actually used by some of the technology programs which the IMAGE geometry supports. The IQUAD option permits the selection of the points to be drawn.

IQUAD = 0	Integer variable which selects the corner points to be drawn.

= 0 Draw input elements.

= 1 Draw picture using quadrilateral element corner points.

The program provides a flexible means of framing the picture data. Because of the flexibility provided, the user should exercise a certain degree of caution in setting up the data to assure the resulting picture will be plotted in the desired location. The following discussion of input variables is designed to help the unfamiliar user in setting up the data for positioning the picture sequences. Figure 9 shows the relationships among the variable input data discussed below.

SCAL = 6. Imaginary frame size in inches for the current picture. All data will be scaled to fit within the specified frame size according to the following relationships.

X-scale factor = (XRG-XLG)/SCAL

Starting X-value = XLG

Y-scale factor = (XRG-XLG)/SCAL

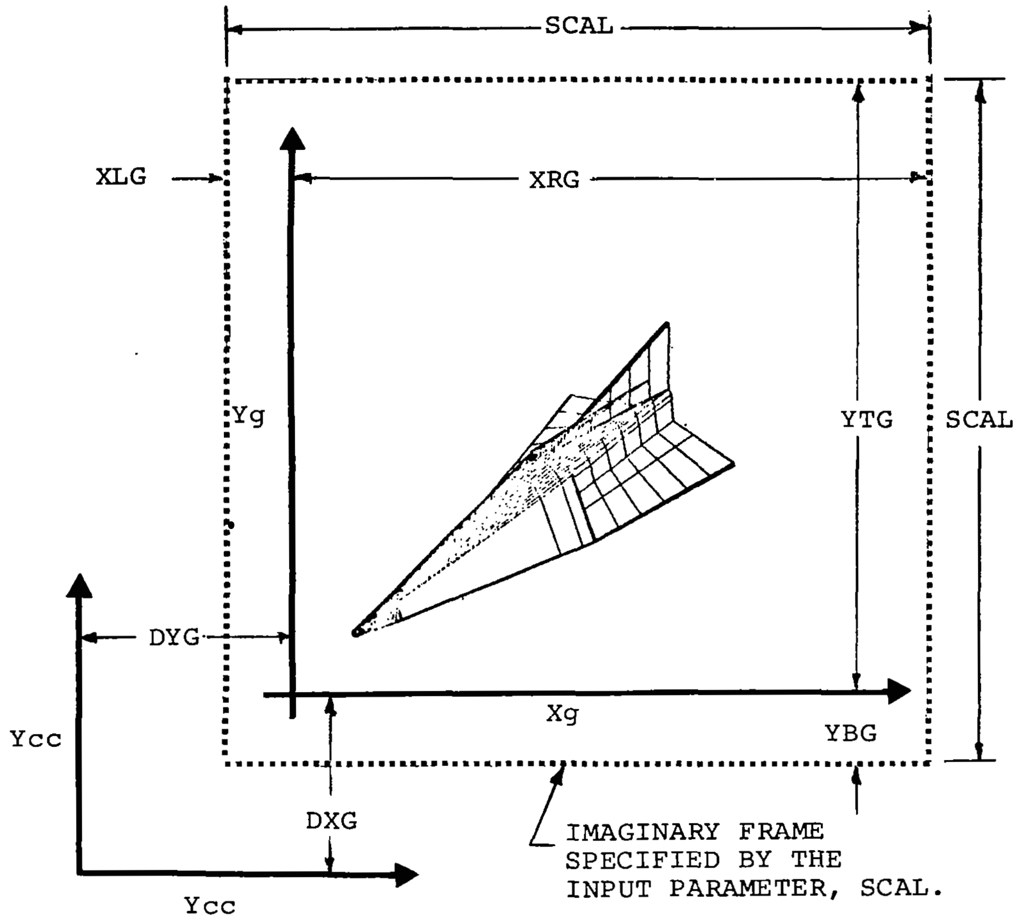
Starting Y-value = XLG

XRG and XLG are computed automatically by the IMAGE program. The coordinate axes used are the result of the initial geometric transformation described above. The relationships used for computing the above parameters internally are as follows:

XLG = min (all X-coordinates)

XRG = max (all X-coordinates)

The above equation generates a "square frame" scaled to the longitudinal configuration geometry dimensions. The "square frame" produces identical scale factors in the X and Y directions. The geometry is quaranteed to fit in the specified frame size. However, the data will not necessarily fit in the frame itself. The position of the picture depends upon the values of the viewing angles. For example, assume the geometry reference is at the nose of the vehicle. The imaginary frame under automatic scaling conditions would encompass the length of the vehicle. Further, the data would all be scaled to the value of the input variable, SCAL. For the above conditions, the position of the nose is controlled by the input variables DXG and DYG.



COORDINATE SYSTEM DEFINITIONS

Xcc, Ycc	Reference coordinates of the plotting device before the current picture is drawn. The initial values are usually specified by the user on the plot request card. Before the picture is drawn they are moved to the Xg, Yg system.
Xg, Yg	Reference axis of the geometric data after the initial transformation specified by the input parameters DELX, DELY, DELZ, XSC, YSC and ZSC.

FIGURE 9 ILLUSTRATION OF THE FRAMING TECHNIQUE EMPLOYED IN THE IMAGE PROGRAM.

Each picture is positioned within the limits of the plotting device using the variables DXG and DYG measured with respect to the previous position.

DXG = 0.	Repositioning of the reference Y-axis in inches for the plotting device before plotting the geometry data for the current picture.
DYG = 0	Repositioning of the reference X-axis in inches for the plotting device before plotting the geometry data for the current picture.

Once the reference axes for the current picture is described, the imaginary frame is defined by the variables YLG, XRG, YBG, YTG. Although computed automatically as described above, the values of XLG, YRG, YBG and YTG may be specified by the user in scaled geometry coordinates with respect to the translated vehicle geometry on TAPE3 as follows:

XLG = computed	Value of the left side of the imaginary frame (geometry scale).
XRG = computed	Value of the right side of the imaginary frame (geometry scale).
YBG = computed	Value of the bottom of the imaginary frame (geometry scale).
YTG = computed	Value of the top of the imaginary frame (geometry scale).

When specifying the above values, the user should remember the "square frame" is essential to the creation of an undistorted picture. Rectangular frames will produce pictures which appear for-shortened in the direction of the short side of the rectangle. For example:

XLG = -10., XRG = 20.,

YBG = -5., YTG = 25.,

would produce a square frame 30 inches on a side. In the above example, the data for the current picture would be scaled as follows:

Y - scale factor = (20.-(-10.))/10. = 3 units/inch

Y - scale factor = (25.-(5.))/10. = 3 units/inch

Starting X-value = -10.

Starting Y-value = -5.

The primary function of the IMAGE program is the generation of pictorial data. Equally important to a geometric description is a description of the vehicle characteristics. The IMAGE program provides the user the option of displaying text information along with the pictorial data. The logical variable controlling this option is TEXT.

TEXT = .FALSE.	Logical variable, if .TRUE., Text information will be read from cards following the $TYP3435 data set in free field format.

In the text option the input variables DXG and DYG position the reference coordinates at the beginning of the first line of text. This reference point will remain until altered by a new $TYP3435 data set. Any number of cards may be read as text. Each card represents a line of text. Lines of text may be skipped by placing a zero (0) in column 1 of the text card. This provides a convenient means of spacing the text information. The text input is terminated by placing the character (2) in column 1 of the last text card. The last card will not be printed.

The maximum number of cards (lines) of text is 53 cards. The character height of the text is controlled by the input variable HTEXT.

HTEXT = 0.14	Height of the characters in the text material.

Post Processing Instructions

The IMAGE program is designed to generate a file of plot commands for a 12-inch vertical height continuous roll paper such as CALCOMP. The file must be on a physical tape for plotting (see figure 3). The user establishes a reference point on the roll at the time the plot request is submitted to be plotted. Generally a one inch offset from the X-axis (Y=1) is adequate. The Y-offset is not applicable since a continuous roll of paper is generally used. A hard coded program header is printed 9.5 inches above the user established reference. All pictures should be scaled and positioned within the 9.5 inch limitation on vertical height, otherwise, the picture may overlay the program header. The horizontal placement of pictures is essentially unlimited.

Externally Generated Geometry

The IMAGE program provides the flexibility of reading the element data for plotting from the normal input device (INPUT) or from an alternate unit. The later capability is provided for the purpose of plotting geometry generated by another computer program.

The information may be stored temporarily or permanently on a system file and attached to the IMAGE execution job step by a method called file substitution.

File substitution is a CDC 6600 system capability which provides a correspondence between internal (logical units) files and external (system) files. The mechanism by which this correspondence is implemented is the "program card." The program card for the IMAGE program is:

```
     PROGRAM IMAGE (INPUT, OUTPUT, TAPE5 = INPUT, TAPE6 =
                    OUTPUT, TAPE3, TAPE 8)
```

In the above illustration the file, TAPE8 is the internal logical unit which may contain the geometric data to be plotted. The correspondence between the internal logical units, TAPE8 and the external file is established by the substitution of the TAPE8 parameter at execution time. For example, assume the data to be plotted was stored on an external file called DATA. The execution card for the IMAGE program would be:

```
     EXECUTE (IMAGE,,,,,,,DATA)
```

During the above execution, the IMAGE program would read from the file called DATA each time the logical unit, TAPE 8, was read. The program card parameters are <u>positional</u>. Therefore, the six commas are essential for the proper use of file substitution, one for each of the other files on the program card.

Use of IMAGE within ODIN

The Optimal Design Integration (ODIN) system is a library of independent computer programs representing the analytical capabilities in a wide variety of technological disciplines. The IMAGE computer program is but a single member of the ODIN library. The sequence of execution of the individual programs is controlled by the executive program, ODINEX (reference 2) which also maintains a name-oriented data base of design information.

Each piece of information is stored by name. The data base forms a communication link among the programs in the library. When used within the ODIN system, IMAGE receives data from the data base before execution. Generally, the ODIN library programs provided information to be stored in the data base. However, IMAGE does not currently generate information of this category.

The actual transfer of information from the data base to IMAGE is performed by ODINEX through pre-processing of the data so the program is "unaware" that it is part of an analysis involving many programs. There are no special input requirements for using IMAGE within the ODIN system. A single control directive,

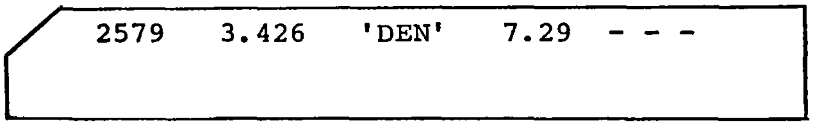
```
'EXECUTE IMAGE'
```

is required for the execution of the program. The delimiter (') is a 4-8 punch. The data which follows this directive is the normal input data described above. However, any data values may come from the data base by specifying the data base name on the input card (in lieu of the actual value).

```
$TYPE32
  :
  :
SCAL = 'SCALE',
  :
  :
$END
```

In the above illustration, the name SCALE is a data base name which may represent a scale factor for the entire configuration. The executive program, ODINEX, replaces the name and the associated delimiter 'SCALE' with the data base value for SCALE. Upon execution of IMAGE, the input component is photometrically scaled by the current scale factor in the data base. A procedure is also available for transferring data base arrays (see reference 2).

The illustration above applies to namelist input, but the procedure for extracting data base information is equally applicable to formatted input. The field width is specified to the position of the delimiters (') as illustrated below:



```
2579    3.426    'DEN'    7.29   - - -
```

DEN is assumed to be the name of a data base variable. The value of DEN will be placed (by ODINEX) on the input card in the most significant (E and F) format left justified in the specified field. If the data base variables were an integer, the value would be right justified in the field (I format).

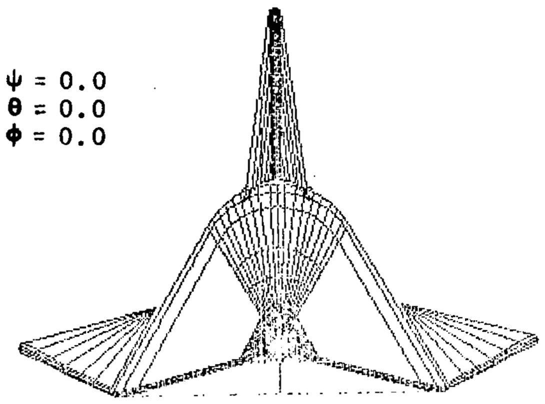
PROGRAM OUTPUT

The output from the IMAGE program is both printed output and plotted output. Geometric characteristics of the quadrilateral elements are available at the user's option. This data is presented through the normal output channels. The plotted data is in the form of a plot tape. The actual plots are obtained by separate submission of a plot request card. Upon submission of the request, the plot tape is processed on the CALCOMP plotting hardware. Pictures of the vehicle at preselected viewing angles, as illustrated in figure 10, may be generated on the CALCOMP device. Input errors can then be corrected before the data is submitted to another ODIN program for the aerodynamic or other technology calculations. Alternately, annotated report quality pictures (see figure 1) may be generated by combining the picture and text options in the program.

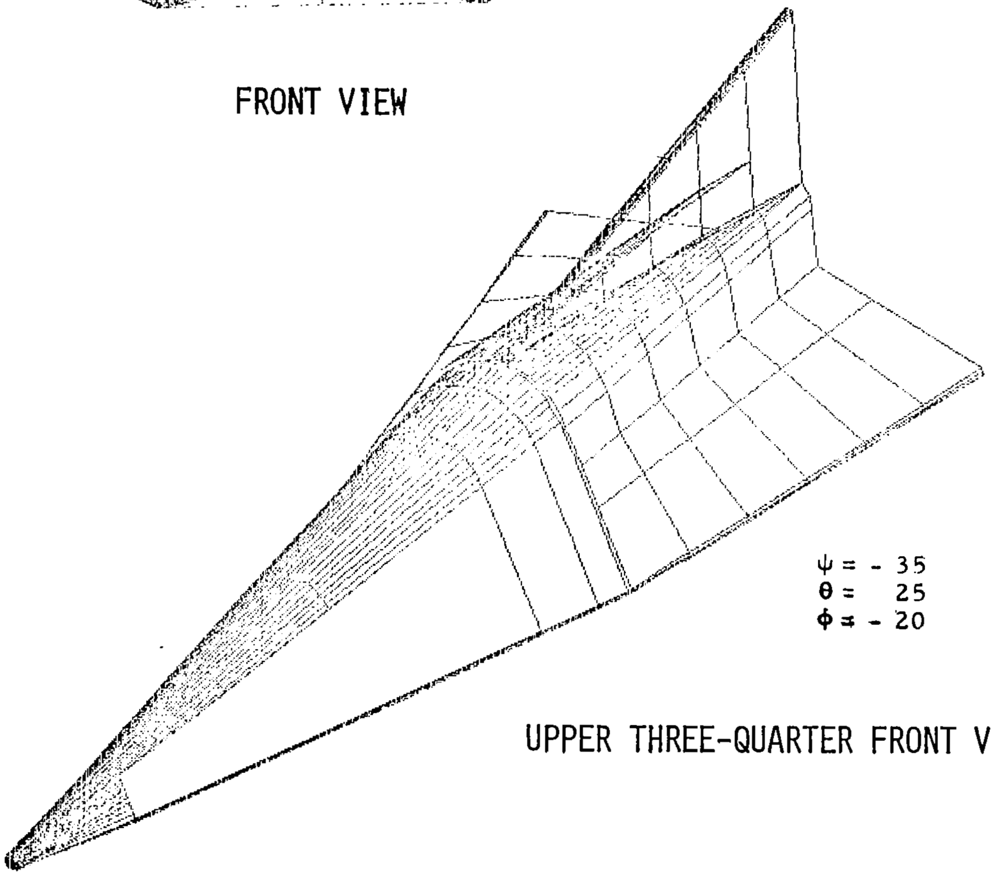
The IMAGE program can also be used to obtain a detailed printout of the properties of each quadrilateral element of the vehicle as illustrated in figure 11. The normal output may also include the accumulated surface area as illustrated in figure 12 and the number of elements for each section of the vehicle. If no print options are specified, only the picture number is printed.

ODIN Output

The current IMAGE program generates no output for the ODIN data base. It has been used primarily for displaying geometric data to be used by other technology programs.

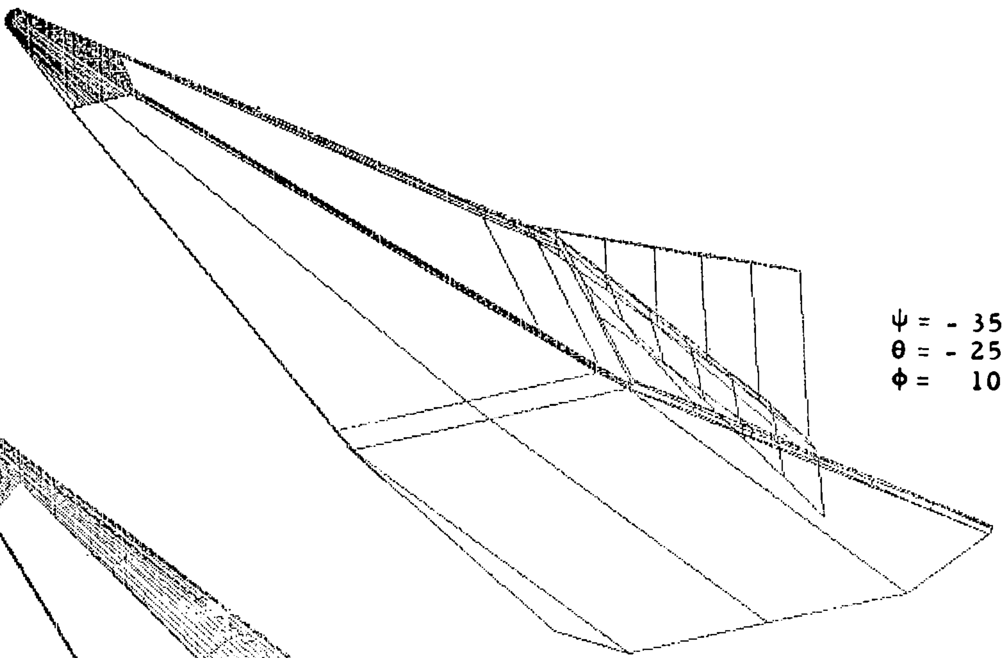


FRONT VIEW

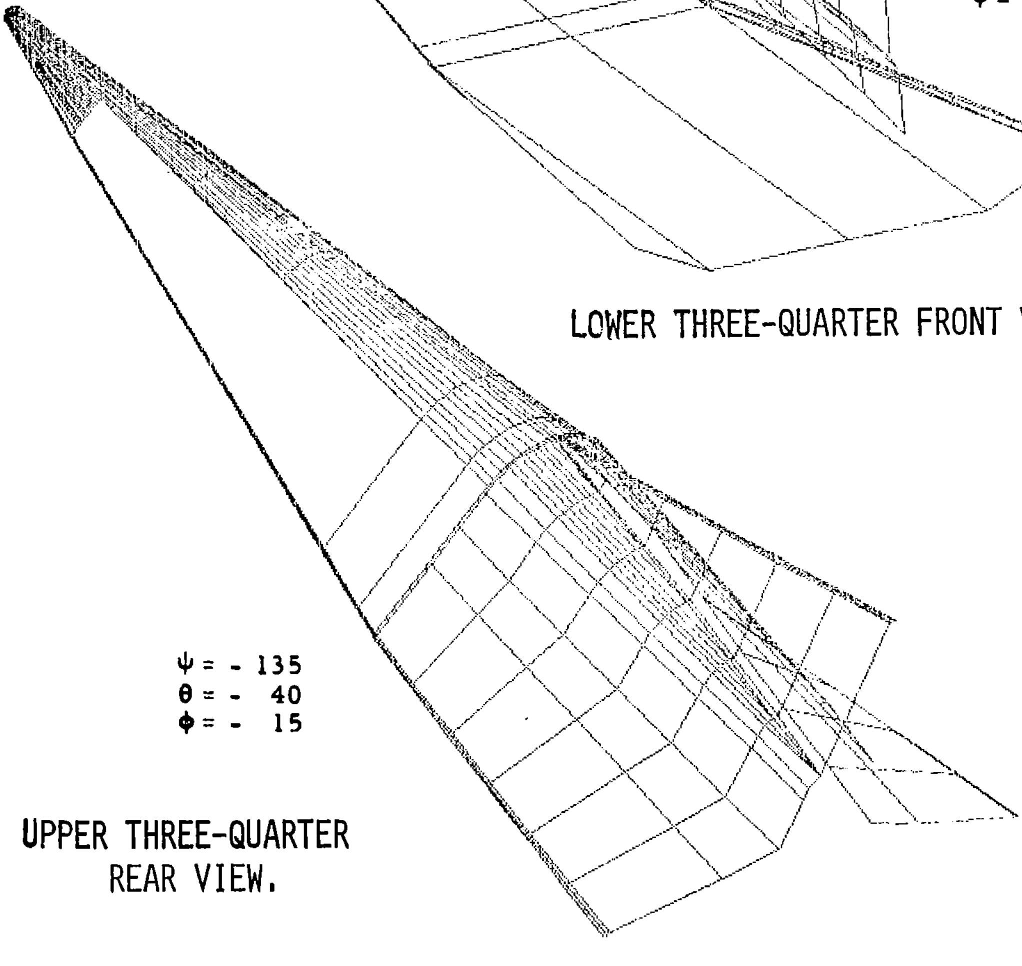


UPPER THREE-QUARTER FRONT VIEW

FIGURE 10A PICTURE OUTPUT FROM IMAGE.



LOWER THREE-QUARTER FRONT VIEW.



UPPER THREE-QUARTER REAR VIEW.

FIGURE 10B PICTURE OUTPUT FROM IMAGE. (CONTINUED)

```
QUADRILATERAL CHARACTERISTICS - PICTURE DRAWING PROGRAM                                    PAGE    0
   CASE    0

INPUT SURFACE ELEMENT DATA
    N    M        X                   X                X                X             NX          XCENT           AREA
                  Y                   Y                Y                Y             NY          YCENT          DELTA V
                  Z                   Z                Z                Z             NZ          ZCENT          VOLUME

     1    1   0.                  0.            -83.00000E-04  -83.00000E-04     .992090  -55.33333E-04   76.52933E-05
              0.                  0.             22.80000E-03   0.               .021691   76.00000E-04   12.61600E-08
              0.                  0.            -62.60000E-03  -66.60000E-03   -.123639  -43.06667E-03   12.61600E-08
          2   0.                  0.            -83.00000E-04  -83.00000E-04     .992088  -55.33333E-04   76.42868E-05
              0.                  0.             42.80000E-03   22.80000E-03     .062987   21.86667E-03   10.52661E-07
              0.                  0.            -51.00000E-03  -62.60000E-03   -.108598  -37.86667E-03   11.78821E-07
          3   0.                  0.            -83.00000E-04  -83.00000E-04     .992088  -55.33333E-04   76.47813E-05
              0.                  0.             57.70000E-03   42.80000E-03     .096047   33.50000E-03   24.60742E-07
              0.                  0.            -33.30000E-03  -51.00000E-03   -.080853  -28.10000E-03   36.39564E-07
          4   0.                  0.            -83.00000E-04  -83.00000E-04     .992093  -55.33333E-04   76.36178E-05
              0.                  0.             65.60000E-03   57.70000E-03     .117932   41.10000E-03   37.01260E-07
              0.                  0.            -11.60000E-03  -33.30000E-03   -.042934  -14.96667E-03   73.40824E-07
          5   0.                  0.            -83.00000E-04  -83.00000E-04     .992091  -55.33333E-04   76.70267E-05
              0.                  0.             65.60000E-03   65.60000E-03     .125524   43.73333E-03   42.10645E-07
              0.                  0.             11.60000E-03  -11.60000E-03    0.000000  -13.87779E-18   11.55147E-06
          6   0.                  0.            -83.00000E-04  -83.00000E-04     .992093  -55.33333E-04   76.36179E-05
              0.                  0.             57.70000E-03   65.60000E-03     .117932   41.10000E-03   37.01260E-07
              0.                  0.             33.30000E-03   11.60000E-03     .042934   14.96667E-03   15.25273E-06
          7   0.                  0.            -83.00000E-04  -83.00000E-04     .992088  -55.33333E-04   76.47813E-05
              0.                  0.             42.80000E-03   57.70000E-03     .096047   33.50000E-03   24.60742E-07
              0.                  0.             51.00000E-03   33.30000E-03     .080853   28.10000E-03   17.71347E-06
          8   0.                  0.            -83.00000E-04  -83.00000E-04     .992088  -55.33333E-04   76.42868E-05
              0.                  0.             22.80000E-03   42.80000E-03     .062987   21.86667E-03   10.52661E-07
              0.                  0.             62.60000E-03   51.00000E-03     .108598   37.86667E-03   18.76613E-06
          9   0.                  0.            -83.00000E-04  -83.00000E-04     .992090  -55.33333E-04   76.52933E-05
              0.                  0.              0.             22.80000E-03     .021691   76.00000E-04   12.61600E-08
              0.                  0.             66.60000E-03   62.60000E-03     .123639   43.06667E-03   18.89229E-06
     2    1 -83.00000E-04  -83.00000E-04  -33.30000E-03  -33.30000E-03     .917135  -22.07026E-03   25.88699E-04
              0.               22.80000E-03   42.80000E-03   0.               .068823   16.90834E-03   24.30574E-07
            -66.60000E-03  -62.60000E-03  -11.75000E-02  -12.50000E-02   -.392589  -95.80336E-03   21.32287E-06
```

FIGURE 11 ILLUSTRATION OF QUADRILATERAL ELEMENT PROPERTIES PRINTOUT FOR IMAGE.

```
 TOTAL AREA OF INPUT ELEMENTS  =         .1925      TOTAL NUMBER OF ELEMENTS =   45
TOTAL VOLUME OF INPUT ELEMENTS =        .016
 TOTAL AREA OF INPUT ELEMENTS  =        2.1657      TOTAL NUMBER OF ELEMENTS =   54
TOTAL VOLUME OF INPUT ELEMENTS =        .093
 TOTAL AREA OF INPUT ELEMENTS  =        4.9149      TOTAL NUMBER OF ELEMENTS =   69
TOTAL VOLUME OF INPUT ELEMENTS =       1.250
 TOTAL AREA OF INPUT ELEMENTS  =        6.0621      TOTAL NUMBER OF ELEMENTS =   78
TOTAL VOLUME OF INPUT ELEMENTS =       1.348
 TOTAL AREA OF INPUT ELEMENTS  =       41.3808      TOTAL NUMBER OF ELEMENTS =   79
TOTAL VOLUME OF INPUT ELEMENTS =       1.348
 TOTAL AREA OF INPUT ELEMENTS  =       76.9257      TOTAL NUMBER OF ELEMENTS =   80
TOTAL VOLUME OF INPUT ELEMENTS =      52.711
 TOTAL AREA OF INPUT ELEMENTS  =       89.7474      TOTAL NUMBER OF ELEMENTS =   82
TOTAL VOLUME OF INPUT ELEMENTS =      81.172
 TOTAL AREA OF INPUT ELEMENTS  =      105.4111      TOTAL NUMBER OF ELEMENTS =  109
TOTAL VOLUME OF INPUT ELEMENTS =      86.445
 TOTAL AREA OF INPUT ELEMENTS  =      111.0823      TOTAL NUMBER OF ELEMENTS =  127
TOTAL VOLUME OF INPUT ELEMENTS =      89.399
 TOTAL AREA OF INPUT ELEMENTS  =      160.1581      TOTAL NUMBER OF ELEMENTS =  130
TOTAL VOLUME OF INPUT ELEMENTS =      89.399
 TOTAL AREA OF INPUT ELEMENTS  =      174.4336      TOTAL NUMBER OF ELEMENTS =  131
TOTAL VOLUME OF INPUT ELEMENTS =     118.473
 TOTAL AREA OF INPUT ELEMENTS  =      204.8862      TOTAL NUMBER OF ELEMENTS =  137
TOTAL VOLUME OF INPUT ELEMENTS =     179.496
 TOTAL AREA OF INPUT ELEMENTS  =      219.9242      TOTAL NUMBER OF ELEMENTS =  143
TOTAL VOLUME OF INPUT ELEMENTS =     198.834
 TOTAL AREA OF INPUT ELEMENTS  =      244.2043      TOTAL NUMBER OF ELEMENTS =  161
TOTAL VOLUME OF INPUT ELEMENTS =     232.146
 TOTAL AREA OF INPUT ELEMENTS  =      254.0524      TOTAL NUMBER OF ELEMENTS =  215
TOTAL VOLUME OF INPUT ELEMENTS =     235.955
 TOTAL AREA OF INPUT ELEMENTS  =      257.2163      TOTAL NUMBER OF ELEMENTS =  230
TOTAL VOLUME OF INPUT ELEMENTS =     240.742
 TOTAL AREA OF INPUT ELEMENTS  =      260.4069      TOTAL NUMBER OF ELEMENTS =  248
TOTAL VOLUME OF INPUT ELEMENTS =     253.090
 TOTAL AREA OF INPUT ELEMENTS  =      286.6821      TOTAL NUMBER OF ELEMENTS =  254
TOTAL VOLUME OF INPUT ELEMENTS =     261.139
 TOTAL AREA OF INPUT ELEMENTS  =      288.2921      TOTAL NUMBER OF ELEMENTS =  272
TOTAL VOLUME OF INPUT ELEMENTS =     261.191
```

FIGURE 12 ILLUSTRATION OF ACCUMULATED SURFACE AREA PRINTOUT FOR IMAGE.

SAMPLE CASES

Usually the most difficult aspect of using any computer program for the first time is mental inertia involved. During the initial learning period, the user gains the necessary confidence required to obtain useful results from the program. The learning period varies with the size and complexity of the particular computer program and the individual involved. The IMAGE program is small and therefore, the learning period should be short.

This section presents four example problems designed to help the first-time user in overcoming the initial mental inertia. Once familiar with the program input, the user will find that providing data to the IMAGE program is not unlike providing data for a configuration layout.

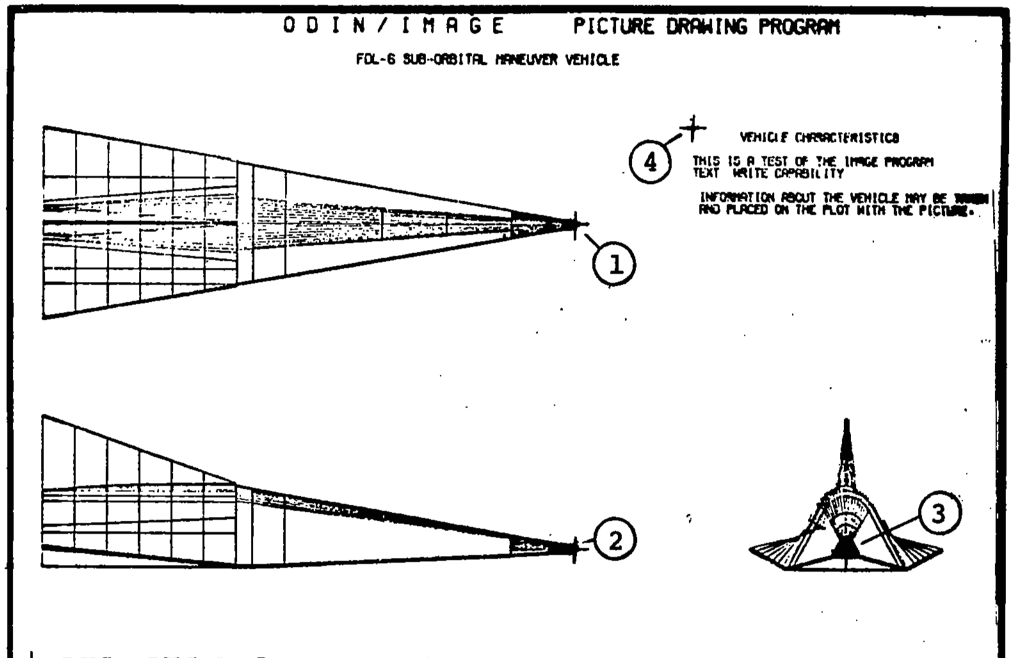
Three-View, Nose Right

The first example is a three-view of a suborbital maneuvering vehicle. The input data for this example is illustrated in figure 12. Also illustrated in figure 13 is an inset drawing resulting from the input data shown. The drawing is reduced several fold from the original 27.9 x 43.2 cm (11 x 17 in) drawing.

Each line of information in figure 13 represents a "card" of input data except as noted in the following discussion. The first card is the title card. This hollerith information appears just below the program heading as illustrated in the inset drawing. The information presented can be up to 59 characters of the TITLE card. The next card is the entire $TYPE32 data set. Note that no data is entered in the data set for this example. Therefore, the program default values are used. By omitting the data entries in this section, the following input data is implied:

PRINTS = 0,	(no printing of quadrilateral characters)
IORIEN = 0,	(normal cross section input)
ISTAT3 = 1,	(one vehicle section)
ITAPE = 0,	(geometric input data in UNIT 5)
IREW8 = 0,	(rewind UNIT 8 before reading - no effect)
XSC = 1.0,	(X - scale factor)

```
FDL-6 SUB-ORBITAL MANEUVER VEHICLE
$TYPE32   $
    [GEOMETRY DATA]
$TYP3435
PSI      = 90..
PHI      = 90..
DXG      = 6..
DYG      = 6..                 (1)
SCAL     = 10..
$
$TYP3435
PHI      = 0..
DXG      = 0..                 (2)
DYG      = -5..
$
$TYP3435
PSI      = 0..
DXG      = 3..                 (3)
DYG      = 0..
$
$TYP3435
TEXT     = .TRUE..
DXG      = -2..                (4)
DYG      =  7..
LAST     = 1.
$
        VEHICLE CHARACTERISTICS
0
THIS IS A TEST OF THE IMAGE PROGRAM
TEXT  WRITE CAPABILITY
0                                                                      Actual Text
 INFORMATION ABOUT THE VEHICLE MAY BE TAKEN FROM THE DATA BASE
 AND PLACED ON THE PLOT WITH THE PICTURE.
2
  END OF PICTURE PHASE
```


ODIN/IMAGE   PICTURE DRAWING PROGRAM
FDL-6 SUB-ORBITAL MANEUVER VEHICLE
VEHICLE CHARACTERISTICS
THIS IS A TEST OF THE IMAGE PROGRAM
TEXT WRITE CAPABILITY
INFORMATION ABOUT THE VEHICLE MAY BE TAKEN
AND PLACED ON THE PLOT WITH THE PICTURE.
1
2
3
4

+ DXG, DYG Reference Points. Illustration Assumes the Vehicle Reference Axes are at the Nose.

FIGURE 13   ILLUSTRATION OF INPUT DATA FOR THREE-VIEW DRAWING - NOSE RIGHT

```
YSC = 1.0,   (Y - scale factor)

ZSC = 1.0,   (Z - scale factor)

DELX = 0.,   (X - translation)

DELY = 0.,   (Y - translation)

DELZ = 0.,   (Z - translation)
```

Even though no data is placed in the $TYPE32 data set, the empty data set shown must be present. The minimum information that must appear in the dummy data set includes the name, the opening $ and closing $ as follows:

```
$TYPE32      $
```

At least one space must appear after the NAMELIST name.

The next set of data is the geometric input data indicated by:

```
GEOMETRY DATA
```

The actual data is omitted from the figure for clarity of presentation. Appendix A is a listing of the actual data used in this and other examples.

The next series of data sets are the $TYP3435 data sets. These data control the position and orientation of a sequence of pictures representing the data described above. The text option is also illustrated. The correspondence between the $TYP3435 data sets and the pictures (text) is indicated by the circled numbers. The crosshairs on the individual drawings refer to the location of the reference system at the start of the individual picture (text). These locations are controlled by DXG and DYG.

The first $TYP3435 data set defines a plan view orientation with the reference axes located 15.2 cm (6 in) to the right and 15.2 cm (6 in) above the initial reference system of the plotting device. The data is to be scaled from the longest X-dimension to fit within a 25.4 cm (10 in) imaginary frame. All other data in this data set will be the default values as follows:

```
THETA = 0        (pitch angle in degrees)

ICS = 0          (connect all four points)
```

IREFL = 0	(draw reflected plane)
ISHAD = 0	(omit drawing rear facing elements)
IAREA = 0	(no section areas will be printed)
IQUAD = 0	(actual corner points will be drawn)
LAST = 0	(this is not the last picture)
XLG = computed	(left side of frame)
XRG = computed	(right side of frame)
YBG = computed	(bottom of frame)
YTG = computed	(top of frame)
TEXT = .FALSE.	(this is a picture option, not text)
XMOVE = 17.0	(move plot device 17 inches in the X-direction after completion of the present picture series)
YMOVE = 0.0	(plot device does not move in the Y-direction after the current picture series)
HTEXT = 0.14	(character height for textual information)

The second $TYP3435 data set defines a profile view of the vehicle. This view is to be located 12.7 cm (5 in) below the plan view (described above). Note that all data remains unchanged between data set definitions. Only those variables requiring change from the previous set need be reset by the user.

The third $TYP3435 data set defines a front view of the vehicle. This view is to be located 7.62 cm (3 in) to the right of the profile view.

The fourth $TYP3435 data set defines a text option, a series of text cards which will be read from cards in 80 column format. Only five input variables have meaning to the text option.

TEXT	(activates text option)
DXG	(X-movement of the plot device reference to the start of the first line of text)

DYG	(Y-movement of the plot device reference to the start of the first line of text)
LAST	(indicates the last $TYP3435 data set in the current series)
HTEXT	(character height for textual information)

The actual text immediately follows the $TYP3435 data set which activates the text option. The first column of each text card is reserved for print control as follows:

0 - skip a line

2 - terminate the text option

In the illustration, the start of the first line of text is positioned 17.8 cm (7 in) above and 5.1 cm (2 in) to the left of the previous (front view) reference and the characters are .36 cm (.14in) high.

The parameter LAST is set to 1 in the text option indicating that current set to be the last $TYP3435 data set in the series. The program flow logic will return for a new title card. Then since the next (title) card has the characters "99" in columns 71 and 72, the IMAGE program execution is terminated.

Three-View, Nose Left

The second example is a three view of a shuttle orbiter-type vehicle. The input data for this example is illustrated in figure 13. The drawing resulting from the data is inset into the figure. Except for the geometric data employed, the primary difference between this example and the previous one is the orientation of the views. This example uses a nose left orientation and a rear view. The previous example used a nose right orientation and a front view. Comparison of figures 12 and 13 will illustrate the differences between the data definition for the two types of vehicle orientation.

The geometric data for this example was generated by a special computer program called PANEL (reference 3). Therefore, the $TYPE32 data set specifies the data source to be external by the parameter.

ITAPE = 1

This parameter specifies the data is to be read from the internal file, UNIT 8. The correspondence between UNIT 8 and the external file which contains the actual data is established through the sixth parameter on the program card. The external source of data is accessed by the IMAGE execution control card through file substitution of the parameter with the external file name as follows:

```
OIMAGE,,,,,GEOM.
```

In the above illustration the file, GEOM, contains the geometric data for the IMAGE program. Of course, the geometric data must have been placed on the GEOM file by the PANEL program in a similar manner as that illustrated above.

```
OPANEL,,,,,GEOM.
```

In the case of the PANEL program, the geometry unit, GEOM, happens to be the fifth file parameter and must have been substituted accordingly at the time PANEL was executed.

The $TYPE32 data set also specifies a translation of the original data in GEOM (TAPE8) 3048 cm (1200 in) forward and 1778 cm (700 inches)down. The reason for the translation is to reposition the reference axis system for plotting purposes. The original data for the orbiter generated in the PANEL program was referenced to the coordinate system of the boost vehicle upon which the orbiter was mounted. The translated reference system is coincidental with the nose of the orbiter as indicated by the inset to figure 14.

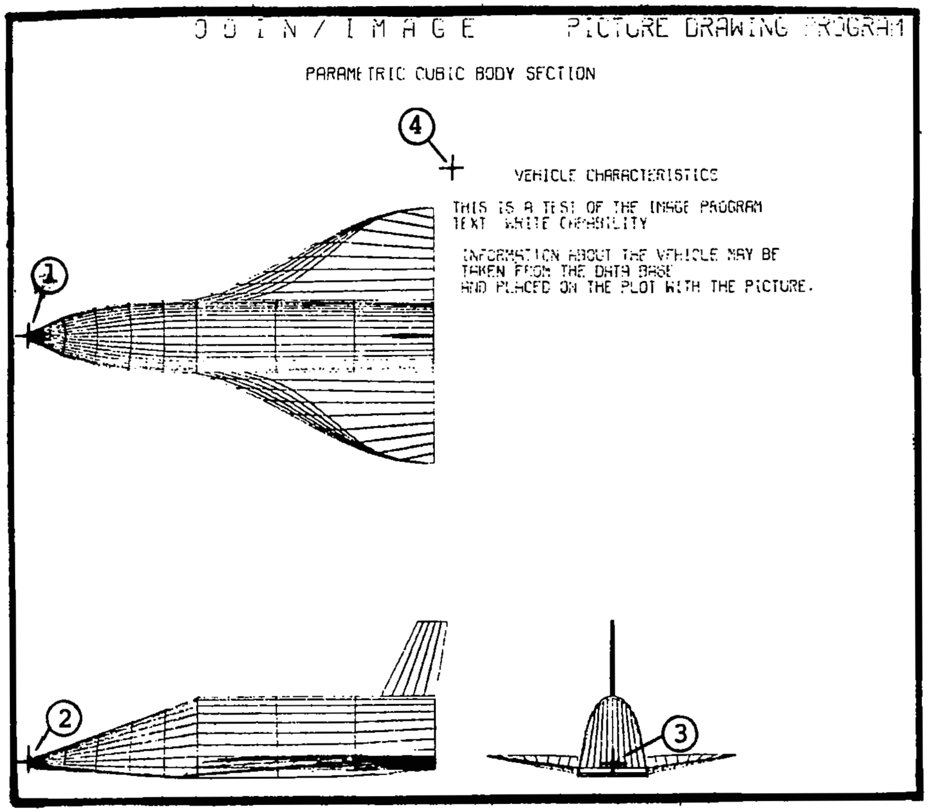
The picture sequence and text option which are defined by the $TYP3435 data sets are similar, except for orientation, to the sequence described in the previous example. Therefore, the reader is referred to that section for discussion of the input data.

Oblique Views

The third example shown in figure 15 illustrates a series of oblique views of the suborbital maneuvering vehicle of example 1. The geometric data is read from the normal input device and not translated as was the data of example 2. The actual data is shown in Appendix A.

Each $TYP3435 data set refers to a numbered view. The correspondence of the individual data set with the view is indicated in the figure. The significant difference between these data sets and the data sets of the other examples are:

```
* PARAMETRIC CUBIC BODY SECTION
$TYPE32   ITAPE=1.
 DELX=1200..
 DELZ=-700..
$
$TYP3435
PSI      = -90..  }
PHI      = -90..  }
DXG      = 0..    }   ①
DYG      = 6..    }
SCAL     = 10..   }
$
$TYP3435
PHI      = 0..    }
DXG      = 0..    }   ②
DYG      = -5..   }
$
$TYP3435
PSI      = 180..  }
DXG      =  7..   }   ③
DYG      = 0..    }
$
$TYP3435
TEXT     = .TRUE..  }
LAST     = 1.       }   ④
DXG      = -2..     }
DYG      =  7..     }
$
        VEHICLE CHARACTERISTICS

THIS IS A TEST OF THE IMAGE PROGRAM           }
TEXT  WRITE CAPABILITY                        }
.                                             }
 INFORMATION ABOUT THE VEHICLE MAY BE         }   ACTUAL TEXT
 TAKEN FROM THE DATA BASE                     }
 AND PLACED ON THE PLOT WITH THE PICTURE.     }
?
  END OF PICTURE PHASE
```



ODIN/IMAGE   PICTURE DRAWING PROGRAM
PARAMETRIC CUBIC BODY SECTION
①
②
③
④

VEHICLE CHARACTERISTICS

THIS IS A TEST OF THE IMAGE PROGRAM
TEXT WRITE CAPABILITY

INFORMATION ABOUT THE VEHICLE MAY BE
TAKEN FROM THE DATA BASE
AND PLACED ON THE PLOT WITH THE PICTURE.

Col. 71-72 → 99

FIGURE 14   ILLUSTRATION OF INPUT DATA FOR THREE-VIEW DRAWING - NOSE LEFT.

```
FDL-6 SUB-ORBITAL MANEUVER VEHICLE
$TYPE32 $

    [GEOMETRIC DATA]

$TYP3435
 PSI = -35.,
THETA = 25.,
 PHI = -20.,
 DXG = 1.,
 DYG = 1.,                 ①
 XLG=0.,
 XRG=33.,
 YBG=0.,
 YTG=33.,
SCAL    = 10.,
$
$TYP3435
 PSI = 0.,
THETA = 0.,
 PHI = 0.,                 ②
DXG     = 1.5,
 DYG = 5.,
$
$TYP3435
 PSI = -35.,
THETA = -25.,
 PHI = 10.,                ③
 DXG = 6.,
 DYG = 2.,
$
$TYP3435
 PSI = -135.,
THETA = -40.,
 PHI =  -15.,              ④
 DYG = -2.,
 DXG = 0.,
 LAST = 1,
$
```

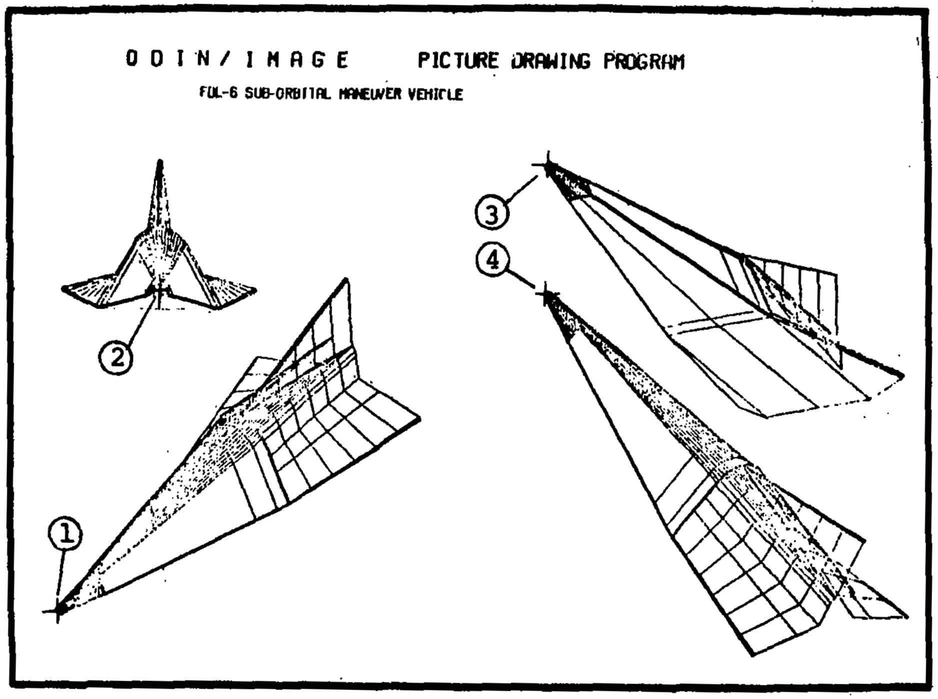


FIGURE 15 ILLUSTRATION OF INPUT DATA FOR A SEQUENCE OF OBLIQUE VIEWS.

1. Differences in viewing angles.
2. Different locations for the views.
3. The absence of text in this example.
4. The frame dimensions XLG, XRG, YBG and YTG are read in rather than computed by the program.

Notice the frame dimensions are established so that the frame size in the X-direction is the same as the frame size in the Y-direction.

XRG - XLG = YTG - YBG

The above criteria establishes a "square frame" which is essential to an undistorted picture.

ODIN Input Example

This example is an illustration of a three-view drawing of a shuttle orbiter recently studied at NASA Langley Research Center using the ODIN procedure to simulate certain aspects of the design process. Figure 16 is an illustration of the data setup for IMAGE within the ODIN framework. This figure portrays a set of data describing a three view, nose left with vehicle characteristics printed on the picture. The resulting picture is inset in the figure. Both the picture definition data and the optional text information is augmented by data base variable names. The variable names are data base interfaces and therefore delimited by (≠). For details of the interface language, the reader is referred to reference 2.

The primary differences between this example and the previous one illustrated in figure 13 are:

1. The present example is an illustration of the program being executed within the ODIN system. The illustrated data was extracted directly from an ODIN simulation.
2. The data is generated by the program of reference 1 which was also executed in the ODIN simulation.
3. The frame size is input from the data base and is based upon the current fuselage length (XLFUS from the data base).

```
 ≠EXECUTE IMAGE≠
≠ADD DM=XLFUS/SCAL≠
 ORBITER DESIGN  25K  P/L        1  INCH= ≠DM    ≠  INCHES
 $TYPE32  ITAPE=1,  ISTAT3=3,  $
 $TYP3435
 PSI     = -90.,
 PHI     = -90.,
  DXG=0.,
  DYG=6.,
  XRG=≠XLFUS ≠,
  XLG=0.,
  YBG=0.,
  YTG=≠XLFUS ≠,
 SCAL    = ≠SCAL ≠,
 $
 $TYP3435
 PHI     = 0.,
 DXG     = 0.,
 DYG     = -5.,
 $
 $TYP3435
 PSI     = 180.,
  DXG=12.,
 DYG     = 0.,
 $
 $TYP3435
 TEXT    = .TRUE.,
 LAST    = 1,
  DXG=-2.5,
  DYG=7.5,
 $
  VEHICLE CHARACTERISTICS
0
 1.  MASS PROPERTIES
0
         LANDED WEIGHT                            ≠WLAND ≠
         ENTRY WEIGHT                             ≠WENTRY≠
         DESIGN CG, SUBSONIC,  25K P/L            ≠XCGTN ≠
         DESIGN CG HYPERSONIC,  13K P/L           ≠XCGHYP≠
0
 2.  GEOMETRY
0
≠ADD DM=XLFUS/12.≠
         BODY LENGTH                              ≠DM    ≠
         TOTAL WING AREA                          ≠STOTAL≠
         CHORD, THEO ROOT                         ≠CROOT ≠
         CHORD, TIP                               ≠CTIP  ≠
         ASPECT RATIO                             ≠AR    ≠
         LEADING EDGE SWEEP ANGLE                 ≠SWTLE ≠
         TRAILING EDGE SWEEP ANGLE                ≠SWTTE ≠
≠ADD DM=CRE+CTE*SPEXP/2.≠
         ELEVON AREA, TOTAL                       ≠DM    ≠
         EXPOSED WING LOCATION                    ≠XOF   ≠
0
 3.  AERODYNAMIC CHARACTERISTICS
0
         DESIGN TRIM LIFT COEFFICIENT             ≠CLT(8)≠
         DESIGN MIMIMUM LANDING SPEED             ≠VMD   ≠
         MAX TRIM ALPHA HYPERSONIC DESIGN COND    ≠ALTRIM≠
2
  END OF PICTURE PHASE
```

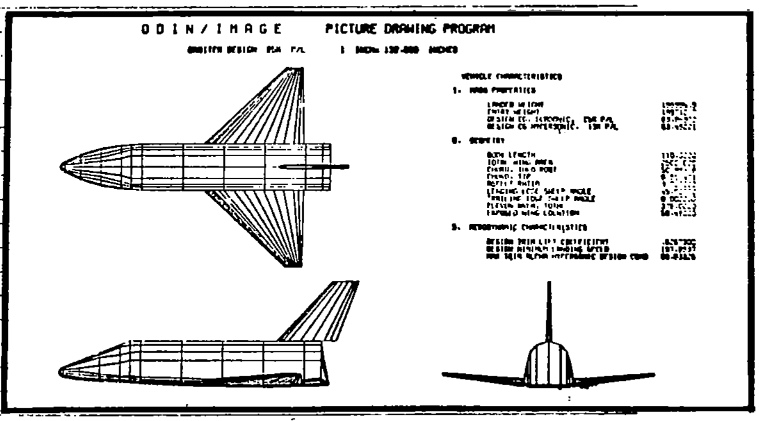


FIGURE 16 ILLUSTRATION OF INPUT DATA FOR IMAGE WHEN USED IN ODIN

4. The text option display actual vehicle characteristics from the data base for the current vehicle. The delimited (≠) names denote the data base names for the indicated quantities.

Each line of information in figure 16 represents a card of input data. The first card illustrated is the ODINEX control directive.

```
≠EXECUTE IMAGE≠
```

The control directive is input information to the ODINEX executive system and accomplishes the following functions:

1. Retrieves the IMAGE program from permanent storage at the beginning of the simulation.

2. Executes the IMAGE program providing the necessary file substitution to pass geometry file to the program from the program of reference 1.

3. Provides for the return of control to the ODINEX executive system after completion of the execution of IMAGE.

The second card is an ADD command which is part of the communication language in the ODINEX executive system described in reference 2. The function of this card,

```
≠ADD DM = XLFUS/SCAL≠
```

is to compute a scale of the drawing (DM) to be used on the picture title card. In the illustration the picture scale is 1 to 132 units. The card illustrated above is "removed from the input stream by ODINEX and is therefore not read by the IMAGE program.

The third card is the TITLE card. This card is generally the first card (when IMAGE is used as an independent program). However, when used with the ODINEX executive system, the TITLE card may not be the first card due to interface requirements with other programs and with the ODIN data base. However, after the input data is preprocessed by ODINEX, the modified input file will be identical in format to the previous examples. No delimited information will appear.

The $TYPE32 data resets the parameter,

```
ITAPE = 1
```

which specifies the geometric data will be obtained from an external source. The ODINEX executive system automatically maintains the proper correspondence between the IMAGE data file and the data files of the program of reference 1. The user need not be concerned with file substitution when using IMAGE with the ODIN system.

Another parameter set in $TYPE32 is:

    ISTAT3=3

This variable specifies that three sections of data on the geometry file will be plotted. Actually more data resided on the geometry file for the example simulation but the first three sections (WING, BODY and VERTICAL TAIL) were the only ones of interest for plotting purposes.

The next series of data sets ($TYP3435) control the position and orientation of a sequence of pictures and text. The first set in the series also defines the frame size and dimensions. Frame size is referenced to the fuselage length, XLFUS and the data is scaled to the parameter, SCAL. Both of these parameters come from the data base as indicated in figure 16.

The last $TYP3435 data set activates the text option. The actual text cards follow the $TYP3435 data set. These cards are modified by data base information as indicated by the delimited variable names. Each delimited name is replaced by the current value of the variable from data base. ODINEX performs this replacement function before the data is read by IMAGE. In this manner, the current vehicle characteristics are extracted from the data base and printed with the geometric representation of the vehicle.

APPENDIX A - PART I

LISTING OF TEXT DATA FOR THE IMAGE PROGRAM

The following pages present a list of the element data for the suborbital maneuvering vehicle test data discussed in connection with the use of the IMAGE program. This sample data is also used in examples contained in Part II of the present report.

```
          0.0000     0.0000    0.00000    0.0000    0.0000    0.00000  -0NOSE 3AERO
          0.0000     0.0000    0.00000    0.0000    0.0000    0.00000  -0NOSE 3AERO
          0.0000     0.0000    0.00000    0.0000    0.0000    0.00000  -0NOSE 3AERO
          0.0000     0.0000    0.00000    0.0000    0.0000    0.00000  -0NOSE 3AERO
          0.0000     0.0000    0.00000    0.0000    0.0000    0.00000  -0NOSE 3AERO
          -.0083     0.0000    -.06661    -.0083     .0228    -.06260  -0NOSE 3AERO
          -.0083      .0428    -.05100    -.0083     .0577    -.03330  -0NOSE 3AERO
          -.0083      .0656    -.01160    -.0083     .0656     .01160  -0NOSE 3AERO
          -.0083      .0577     .03330    -.0083     .0428     .05100  -0NOSE 3AERO
          -.0083      .0228     .06260    -.0083    -.0000     .06660  -0NOSE 3AERO
          -.0333     0.0000    -.12501    -.0333     .0428    -.11750  -0NOSE 3AERO
          -.0333      .0803    -.04580    -.0333     .1083    -.06250  -0NOSE 3AERO
          -.0333      .1231    -.02170    -.0333     .1231     .02170  -0NOSE 3AERO
          -.0333      .1083     .06250    -.0333     .0803     .09580  -0NOSE 3AERO
          -.0333      .0428     .11750    -.0333    -.0000     .12500  -0NOSE 3AERO
          -.0750     0.0000    -.17901    -.0750     .0612    -.16820  -0NOSE 3AERO
          -.0750      .1151    -.13710    -.0750     .1550    -.08950  -0NOSE 3AERO
          -.0750      .1763    -.03110    -.0750     .1763     .03110  -0NOSE 3AERO
          -.0750      .1551     .08950    -.0750     .1151     .13710  -0NOSE 3AERO
          -.0750      .0612     .16820    -.0750    -.0000     .17900  -0NOSE 3AERO
          -.1625     0.0000    -.23601    -.1625     .0807    -.22180  -0NOSE 3AERO
          -.1625      .1517    -.18080    -.1625     .2044    -.11800  -0NOSE 3AERO
          -.1625      .2324    -.04100    -.1625     .2324     .04100  -0NOSE 3AERO
          -.1625      .2044     .11800    -.1625     .1517     .18080  -0NOSE 3AERO
          -.1625      .0807     .22180    -.1625    -.0000     .23600  -0NOSE 3AERO
          -.2500     0.0000    -.25001    -.2500     .0855    -.23490  -0NOSE 3AERO
          -.2500      .1607    -.19150    -.2500     .2165    -.12500  -0NOSE 3AERO
          -.2500      .2462    -.04340    -.2500     .2462     .04340  -0NOSE 3AERO
          -.2500      .2165     .12500    -.2500     .1607     .19150  -0NOSE 3AERO
          -.2500      .0855     .23490    -.2500    -.0000     .25000  -0NOSE 3AERO
     -.25         0.0       -.25        2-.25        .02       -.249          NEBR 3AERO
     -.25          .05      -.245        -.25        .08       -.238          NEBR 3AERO
     -.25          .12      -.22         -.25        .15       -.20           NEBR 3AERO
     -.25          .17      -.195        -.25        .19       -.16           NEBR 3AERO
     -.25          .205     -.14         -.25        .22       -.12           NEBR 3AERO
     -4.          0.0       -.45        1-4.         .04       -.45           NEBR 3AERO
     -4.           .09      -.45         -4.         .15       -.45           NEBR 3AERO
     -4.           .24      -.45         -4.         .34       -.45           NEBR 3AERO
     -4.           .41      -.45         -4.         .53       -.45           NEBR 3AERO
```

```
-4.        .66        -.45       -4.        .82        -.45        NFBB  3AERO
-.25       .23        -.34      2-.25       .24        -.06        NFBT  3AERO
-.25       .245       -.035      -.25       .248       -.02        NFBT  3AERO
-.25       .250       0.         -.25       .248       .02         NFBT  3AERO
-.25       .245       .045       -.25       .243       .06         NFBT  3AERO
-.25       .235       .07        -.25       .22        .12         NFBT  3AERO
-.25       .21        .10        -.25       .19        .165        NFBT  3AERO
-.25       .19        .15        -.25       .165       .19         NFBT  3AERO
-.25       .15        .205       -.25       .13        .22         NFBT  3AERO
-4.        .80        -.34      1-4.        .83        -.22        NFBT  3AERO
-4.        .73        -.11       -4.        .75        -.06        NFBT  3AERO
-4.        .72        [illegible] -4.       .70        .04         NFBT  3AERO
-4.        .66        .17        -4.        .63        .17         NFBT  3AERO
-4.        .60        .24        -4.        .56        .31         NFBT  3AERO
-4.        .54        .30        -4.        .51        .42         NFBT  3AERO
-4.        .48        .44        -4.        .45        .54         NFBT  3AERO
-4.        .42        .60        -4.        .39        .67         NFBT  3AERO
-.25       .13        .22       2-.25       .12        .225        NFBT  3AERO
-.25       .11        .235       -.25       .09        .24         NFBT  3AERO
-.25       .075       .243       -.25       .06        .244        NFBT  3AERO
-.25       .05        .245       -.25       .035       .247        NFBT  3AERO
-.25       .02        .249       -.25       0.0        .250        NFBT  3AERO
-4.        .39        .67       1-4.        .36        .72         NFBT  3AERO
-4.        .32        .76        -4.        .29        .79         NFBT  3AERO
-4.        .24        .83        -4.        .20        .85         NFBT  3AERO
-4.        .155       .87        -4.        .11        .89         NFBT  3AERO
-4.        .05        .895       -4.        0.0        .90         NFBT  3AERO
-4.        0.0        -.45      2-4.        .82        -.45        FBB   3AERO
-20.       0.0        -1.2      1-20.       3.59       -1.2         FBB   3AERO
-4.        .89        -.34      2-4.        .39        .67         FBS   3AERO
-18.       3.29       -.99      1-18.       1.41       2.44         FBS   3AERO
-18.       3.29       -.99      2-18.       1.41       2.44         FBS   3AERO
-20.       3.645      -1.1      1-20.       1.58       2.74         FBS   3AERO
-21.       3.83       -1.1      1-21.       1.64       2.84         FBS   3AERO
-4.        .39        .67       2-4.        .36        .72         FBT   3AERO
-4.        .32        .70        -4.        .29        .79         FBT   3AERO
-4.        .24        .83        -4.        .20        .85         FBT   3AERO
-4.        .155       .87        -4.        .11        .89         FBT   3AERO
-4.        .05        .895       -4.        0.0        .90         FBT   3AERO
```

```
-8.        .72      1.23     1-8.       .66     1.39         FBT  3AERO
-8.        .59      1.38      -8.       .52     1.43         FBT  3AERO
-8.        .43      1.49      -8.       .36     1.52         FBT  3AERO
-8.        .275     1.56      -8.       .19     1.58         FBT  3AERO
-8.        .12      1.595     -8.       0.0     1.60         FBT  3AERO
-12.       1.1      1.74     1-12.      .93     1.84         FBT  3AERO
-12.       .83      1.95      -12.      .74     2.04         FBT  3AERO
-12.       .62      2.12      -12.      .52     2.18         FBT  3AERO
-12.       .39      2.24      -12.      .25     2.27         FBT  3AERO
-12.       .11      2.295     -12.      0.0     2.3          FBT  3AERO
-18.       1.4      2.44     1-18.      1.31    2.6          FBT  3AERO
-18.       1.18     2.77      -18.      1.06    2.89         FBT  3AERO
-18.       .88      3.03      -18.      .74     3.115        FBT  3AERO
-18.       .56      3.2       -18.      .38     3.25         FBT  3AERO
-18.       .15      3.29      -18.      0.0     3.3          FBT  3AERO
-18.       1.4      2.44     2-18.      1.31    2.6          FBT  3AERO
-18.       1.18     2.77      -18.      1.06    2.89         FBT  3AERO
-18.       .98      3.03      -18.      .74     3.115        FBT  3AERO
-18.       .56      3.2       -18.      .38     3.25         FBT  3AERO
-18.       .15      3.29      -18.      0.0     3.3          FBT  3AERO
-20.       1.58     2.74     1-20.      1.47    2.905        FBT  3AERO
-20.       1.31     3.09      -20.      1.17    3.22         FBT  3AERO
-20.       .98      3.36      -20.      .82     3.45         FBT  3AERO
-20.       .62      3.54      -20.      .42     3.60         FBT  3AERO
-20.       .16      3.638     -20.      0.0     3.64         FBT  3AERO
-21.       1.64     2.84     1-21.      1.53    3.02         FBT  3AERO
-21.       1.37     3.23      -21.      1.23    3.37         FBT  3AERO
-21.       1.03     3.53      -21.      .86     3.63         FBT  3AERO
-21.       .65      3.725     -21.      .44     3.79         FBT  3AERO
-21.       .17      3.84      -21.      0.0     3.85         FBT  3AERO
-19.99     3.0      -1.1995  2-19.99    3.58    -1.1995      ABB  3AERO
-20.       3.0      -1.2     1-20.      3.58    -1.2         ABB  3AERO
-21.       3.0      -1.2     1-21.      3.78    -1.2         ABB  3AERO
-33.       3.0      -1.2     1-33.      3.78    -1.2         ABB  3AERO
-21.       3.78     -1.2     2-21.      3.78    -1.2         CSS  3AERO
-33.       3.78     -1.2     1-33.      5.90    -.12         CSS  3AERO
-21.       3.90     -1.01    2-21.      2.84    .90          ABO  3AERO
-23.       4.24     -.81     1-23.      2.84    .90          ABO  3AERO
-25.       4.56     -.61     1-25.      2.84    .90          ABO  3AERO
```

-27.	4.9[illegible]	-.4[illegible]	1-27.	2.84	.90	ABO 3AERO
-29.	5.24	-.2[illegible]	1-29.	2.84	.90	ABO 3AERO
-31.	5.58	-.025	1-31.	2.84	.90	ABO 3AERO
-33.	5.09	.13	1-33.	2.84	.90	ABO 3AERO
-21.	2.84	.9[illegible]	2-21.	2.33	1.87	ABO 3AERO
-23.	2.84	.9[illegible]	1-23.	2.16	1.76	ABO 3AERO
-25.	2.84	.9	1-25.	2.0	1.67	ABO 3AERO
-27.	2.84	.9[illegible]	1-27.	1.83	1.58	ABO 3AERO
-29.	2.84	.9[illegible]	1-29.	1.66	1.48	ABO 3AERO
-31.	2.84	.9	1-31.	1.50	1.40	ABO 3AERO
-33.	2.84	.9[illegible]	1-33.	1.33	1.32	ABO 3AERO
-21.	2.33	1.07	2-21.	1.82	2.8	ABST 3AERO
-21.	1.59	3.15	-21.	1.33	3.44	ABST 3AERO
-23.	2.16	1.76	1-23.	1.7	2.8	ABST 3AERO
-23.	1.51	3.15	-23.	1.27	3.44	ABST 3AERO
-25.	2.[illegible]	1.[illegible]7	1-25.	1.58	2.8	ABST 3AERO
-25.	1.42	3.[illegible]	-25.	1.20	3.44	ABST 3AERO
-27.	1.83	1.[illegible]8	1-27.	1.45	2.8	ABST 3AERO
-27.	1.32	3.15	-27.	1.12	3.44	ABST 3AERO
-29.	1.66	1.46	1-29.	1.3	2.8	ABST 3AERO
-29.	1.19	3.15	-29.	1.02	3.44	ABST 3AERO
-31.	1.5[illegible]	1.4[illegible]	1-31.	1.15	2.8	ABST 3AERO
-31.	1.[illegible]7	3.15	-31.	0.93	3.44	ABST 3AERO
-33.	1.33	1.32	1-33.	1.05	2.8	ABST 3AERO
-33.	[illegible].98	3.15	-33.	0.74	3.44	ABST 3AERO
-21.	1.33	3.44	2-21.	1.2	3.55	ABST 3AERO
-21.	1.[illegible]5	3.[illegible]5	-21.	0.90	3.73	ABST 3AERO
-21.	0.75	3.[illegible]	-21.	0.60	3.86	ABST 3AERO
-21.	0.45	3.[illegible]	-21.	0.30	3.93	ABST 3AERO
-21.	0.15	3.[illegible]	-21.	0.0	3.96	ABST 3AERO
-23.	1.27	3.44	1-23.	1.13	3.55	ABST 3AERO
-23.	0.94	3.[illegible]	-23.	0.85	3.73	ABST 3AERO
-23.	0.72	3.[illegible]	-23.	0.57	3.86	ABST 3AERO
-23.	0.44	3.[illegible]	-23.	0.23	3.93	ABST 3AERO
-23.	0.23	3.43	-23.	0.23	3.93	ABST 3AERO
-25.	1.2[illegible]	3.44	1-25.	1.08	3.55	ABST 3AERO
-25.	0.95	3.[illegible]	-25.	0.82	3.73	ABST 3AERO
-25.	[illegible].67	3.[illegible]	-25.	0.53	3.86	ABST 3AERO
-25.	[illegible].37	3.[illegible]	-25.	0.37	3.9	ABST 3AERO

```
-25.          0.37       3.9      -25.        0.29       3.9              ABST 3AERO
-27.          1.12       3.44    1-27.        1.02       3.55             ABST 3AERO
-27.          0.93       3.65     -27.        0.82       3.73             ABST 3AERO
-27.          0.68       3.8      -27.        0.49       3.86             ABST 3AERO
-27.          0.49       3.86     -27.        0.49       3.86             ABST 3AERO
-27.          0.49       3.86     -27.        0.39       3.86             ABST 3AERO
-29.          1.02       3.44    1-29.        0.93       3.55             ABST 3AERO
-29.          0.83       3.65     -29.        0.72       3.72             ABST 3AERO
-29.          0.63       3.78     -29.        0.63       3.78             ABST 3AERO
-29.          0.63       3.78     -29.        0.63       3.78             ABST 3AERO
-29.          0.63       3.78     -29.        0.49       3.78             ABST 3AERO
-31.          0.93       3.44    1-31.        0.86       3.55             ABST 3AERO
-31.          0.77       3.64     -31.        0.77       3.64             ABST 3AERO
-31.          0.77       3.64     -31.        0.77       3.64             ABST 3AERO
-31.          0.77       3.64     -31.        0.77       3.64             ABST 3AERO
-31.          0.77       3.64     -31.        0.61       3.64             ABST 3AERO
-33.          0.74       3.44    1-33.        0.74       3.44             ABST 3AERO
-33.          0.74       3.44     -33.        0.74       3.44             ABST 3AERO
-33.          0.74       3.44     -33.        0.74       3.44             ABST 3AERO
-33.          0.74       3.44     -33.        0.74       3.44             ABST 3AERO
-33.          0.74       3.44     -33.        0.74       3.44             ABST 3AERO
     -0.2500     0.2191   -0.12042  -0.2500      0.2225  -0.11400      BLF  3AERO
     -0.2500     0.2257   -0.10750  -0.2500      0.2287  -0.10100      BLF  3AERO
     -0.2500     0.2315   -0.09430  -0.2500      0.2342  -0.08760      BLE  3AERO
     -4.0000     0.8200   -0.45831  -4.0000      0.8519  -0.45200      BLE  3AERO
     -4.0000     0.8789   -0.43390  -4.0000      0.8970  -0.40690      BLE  3AERO
     -4.0000     0.9033   -0.37500  -4.0000      0.8970  -0.34310      BLE  3AERO
    -20.0000     3.5800   -1.20001 -20.0000      3.6119  -1.19370      BLE  3AERO
    -20.0000     3.6389   -1.17560 -20.0000      3.6570  -1.14860      BLE  3AERO
    -20.0000     3.6633   -1.11670 -20.0000      3.6570  -1.0848       BLE  3AERO
    -21.0000     3.7600   -1.20001 -21.0000      3.7919  -1.19370      BLE  3AERO
    -21.0000     3.8189   -1.17560 -21.0000      3.8370  -1.14860      BLE  3AERO
    -21.0000     3.8433   -1.11670 -21.0000      3.8370  -1.08480      BLE  3AERO
-21.          3.78      -1.2     2-21.        3.93      -1.12             BLE  3AERO
-21.          3.93      -1.05     -21.        3.92      -1.01             BLE  3AERO
-23.          4.21      -1.01    1-23.        4.28      -.92              BLE  3AERO
-23.          4.29      -.85      -23.        4.24      -.91              BLE  3AERO
-25.          4.55      -.82     1-25.        4.62      -.76              BLE  3AERO
-25.          4.62      -.67      -25.        4.56      -.63              BLF  3AERO
```

```
-27.        4.92       -.6[illegible]      1-27.       4.9[illegible]      -.[illegible]        BLE   3AERO
-27.        4.95       -.4         -27.       4.92      -.43        BLE   3AERO
-29.        5.24       -.46       1-29.       5.31      -.27        BLE   3AERO
-29.        5.30       -.3         -29.       5.24      -.25        BLE   3AERO
-31.        5.57       -.28       1-31.       5.66      -.18        BLE   3AERO
-31.        5.65       -.17        -31.       5.58      -.055       BLE   3AERO
-33.        5.91       -.11       1-33.       6.00       0.0        BLE   3AERO
-33.        5.98        .08        -33.       5.89       .13        BLE   3AERO
-21.        0.1         3.95      2-21.       0.1        3.95       VFN   3AERO
-23.        0.20        [illegible]      1-23.       0.1        4.6        VFN   3AERO
-25.        0.29        [illegible]      1-25.       0.11       5.24       VFN   3AERO
-27.        0.39        3.00      1-27.       0.11       5.95       VFN   3AERO
-29.        0.49        3.78      1-29.       0.11       6.54       VFN   3AERO
-31.        0.61        3.64      1-31.       0.11       7.34       VFN   3AERO
-33.        0.74        [illegible]      1-33.       0.11       8.03       VFN   3AERO
-21.        0.1         [illegible]      2-21.       0.07       3.95       FNLE  3AERO
-21.        0.04        [illegible]       -21.       [illegible]       3.95       FNLF  3AERO
-23.        0.1         4.5       1-23.       0.07       4.63       FNLF  3AERO
-23.        0.04        4.64       -23.       0.0        4.65       FNLF  3AERO
-25.        0.11        5.29      1-25.       0.07       5.33       FNLE  3AERO
-25.        0.04        5.34       -25.       0.0        5.35       FNLE  3AFRO
-27.        0.11        [illegible]      1-27.       0.07       6.0        FNLE  3AERO
-27.        0.04        [illegible]       -27.       0.0        6.02       FNLF  3AERO
-29.        0.11        [illegible]      1-29.       0.07       6.69       FNLF  3AFRO
-29.        0.04        [illegible].71     -29.       0.0        6.72       FNLF  3AERO
-31.        0.11        7.34      1-31.       0.07       7.35       FNLF  3AERO
-31.        0.04        7.41       -31.       0.0        7.43       FNLF  3AERO
-33.        0.11        8.[illegible]3     1-33.       0.07       8.05       FNLE  3AERO
-33.        0.04        [illegible]       -33.       0.0        8.08       FNLF  3AERO
```

PART II - PROGRAM HIDDEN

Program HIDDEN provides a true hidden line picture drawing capability for the ODIN system. This program differs from IMAGE both in computer time requirements and fidelity of the picture. HIDDEN requires significant amounts of computer time for execution but provides an accurate rendition of complex geometries. IMAGE, on the other hand, requires little computer time but only provides an accurate picture for convex objects. Thus, the combination of programs provides either accurate/ time consuming pictures or rough first cut/rapid running pictures. Choice of programs in an ODIN simulation therefore rests with the analyst and his requirements.

Program HIDDEN is an extended version of a program originally written at Purdue University. The present authors wish to acknowledge their debt to this source at this point. Use of existing codes is of course a major feature of the ODIN system. In fact, the ability to incorporate existing codes in the ODIN system is the underlying reason for the order of magnitude reduction in program costs when ODIN is compared to other major vehicle design synthesis programs which operate or seek to operate at the technology levels employed in current ODIN simulations.

The description of Program HIDDEN which follows is an edited version of the Purdue University final report of Grant NGR-15-05-180 augmented by a description of the procedures used to generate pictures applicable to ODIN simulations. When used in the PCSYS picture drawing system HIDDEN operates with two additional programs written by Aerophysics Research Corporation. These programs are CVRTHD which automatically converts IMAGE input to HIDDEN input and PLOTHD which converts HIDDEN output to a form acceptable to the Langley Research Center and Aerophysics Research Corporation plotting programs. Program CVRTHD and PLOTHD are described in Part III of the present report.

PROGRAM DESCRIPTION

Coordinate System

Program HIDDEN input consists of a list of points and a manner for specifying the connectivity of these points to form an object. All objects can be reasonably well represented by lines, planes, and holes in planes. Thus, the most basic input scheme for a true hidden line algorithm requires a means of inputing points and then describing which points formed planes, holes, and lines.

The specification of three dimensional points required the adoption of a reference coordinate system. The right handed coordinate system as shown in Figure 17 was adopted. It should be noted that this convention differs by a rotation from that of program IMAGE on page 3.

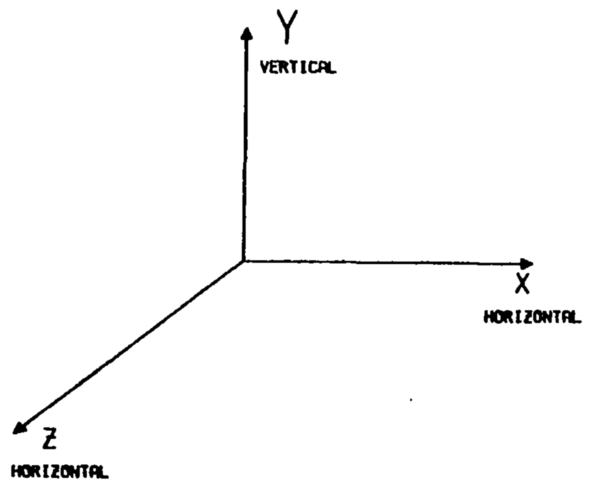


FIGURE 17. BASIC COORDINATE SYSTEM

The positive Y-axis can be thought of as being vertical and aimed upward and the X-Z plane to be the horizontal or ground plane. These statements are based on the viewing and display transformations present in the Hidden Line Removal Program.

Surface Model Description

When deciding the inputs to enter, a dominant goal was to keep the required data to a minimum and to make the input as "natural" as possible. To make the input "natural" Purdue decided to use the words "POINTS", "LINES", "HOLES", "PLANES", and "END" to indicate the type of data following. These words are referred to as keywords. Each keyword sets the data type which is to follow and that data type remains in effect until another keyword is encountered. Thus, after the occurrence of the keyword "PLANES" several planes may be entered without each being preceded by the word "PLANES".

If the planes, holes, and lines are to be described by references to points which have been previously read or are to be read, some scheme for the identification of the points must be included in the input routine. An obvious scheme is to identify each point with a number. The computer numbers the points sequentially, beginning at 1, as it encounters them. The user, although not required to enter these numbers, must be aware of them in order to enter the planes, holes, and lines. This solution avoids the problems arising out of the less ordered numbers a user might supply. It also saves some data preparation time.

With each point assigned a sequence number all the user specifies a plane or hole by listing the sequence numbers assigned to the vertices forming the plane or hole. The sequence numbers must be entered in order proceeding around the surface element boundary. There is no restriction on the direction in which the boundary is traveled. Many algorithms require that the vertices be entered so that if vectors were formed from vertex 1 to 2, 2 to 3, etc., and two sequential vectors are crossed that an outward normal would be generated. This requirement can be awkward for some three dimensional object input requirements. Program IMAGE avoids this problem by virture of other programs in the ODIN system which automatically prepare the surface geometries. Program PANEL, reference 6, is an example of one such program. In program HIDDEN the list of vertices forming planes or holes is completed with the second occurrence of the first vertex. Each new plane, hole, or line begines on a new line or card and, in the case of planes and holes, if there are more vertices than room permits the list is simply continued onto the next line or card.

If there exists other lines in the desired object besides those forming the plane and hole boundaries, their endpoint sequence numbers are given following the entry of a "LINE" keyward. The line feature permits the decoration or lettering of plane surfaces as well as the inclusion of such elements as ropes, wires, cables, etc. into three dimensional objects.

Restrictions on data input are that holes must immediately follow the plane in which they occur and that after all data describing an object has been given an "END" keyword should be given.

Data Formats

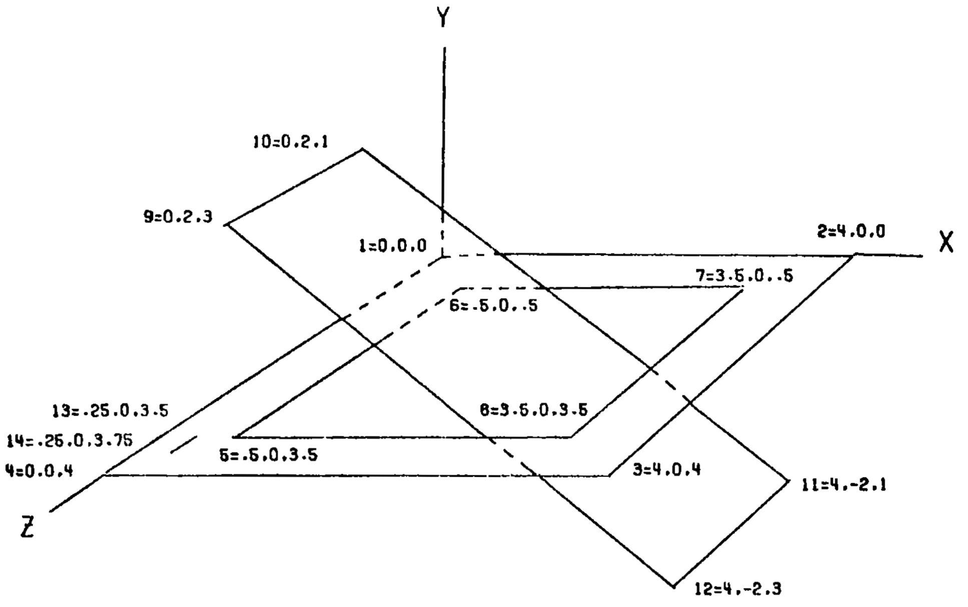
All existing implementations of this basic input scheme have been done in FORTRAN. The format for the various pieces of data are as follows:

KEYWORDS	A3,77X
PLANES	8X,1814
HOLES	8X,1814
LINES	8X,214
POINTS	10X,3F10.3

An example is given in Figure 18 showing a simple object and the required data input.

Program Operation

As can be seen by the flow diagram of Figure 19, the program begins by asking the user to assign three files to logical unit numbers. Two of the three are for input and the remaining file is for output. One input file contains the object description using either the basic input or the Composite Object Input Language. The other input file contains the viewing vector for each view. The viewing vector consists of two points given relative to the object coordinate system which denote where the observer is standing and where he is looking. The only restriction on this vector is that the observer must be outside the object because no clipping is provided. The third file, the output file, is that file which is written with the lines to be drawn. This output file can then be used as input to programs producing displays on suitable graphics display devices.



```
COLUMN NUMBER:
1        1         2          3
         1         1          1

POINTS
         0.        0.         0.
         4.        0.         0.
         4.        0.         4.
         0.        0.         4.
         .5        0.         3.5
         .5        0.         .5
         3.5       0.         .5
         3.5       0.         3.5
         0.        2.         3.
         0.        2.         1.
         4.        -2.        1.
         4.        -2.        3.
PLANES
           9   10   11   12   9
           1    2    3    4   1
HOLE
           5    6    7    8    5
POINTS
         .25       0.         3.5
         .25       0.         3.75
LINES
         13   14
END
```

FIGURE 18. EXAMPLE OF BASIC INPUT

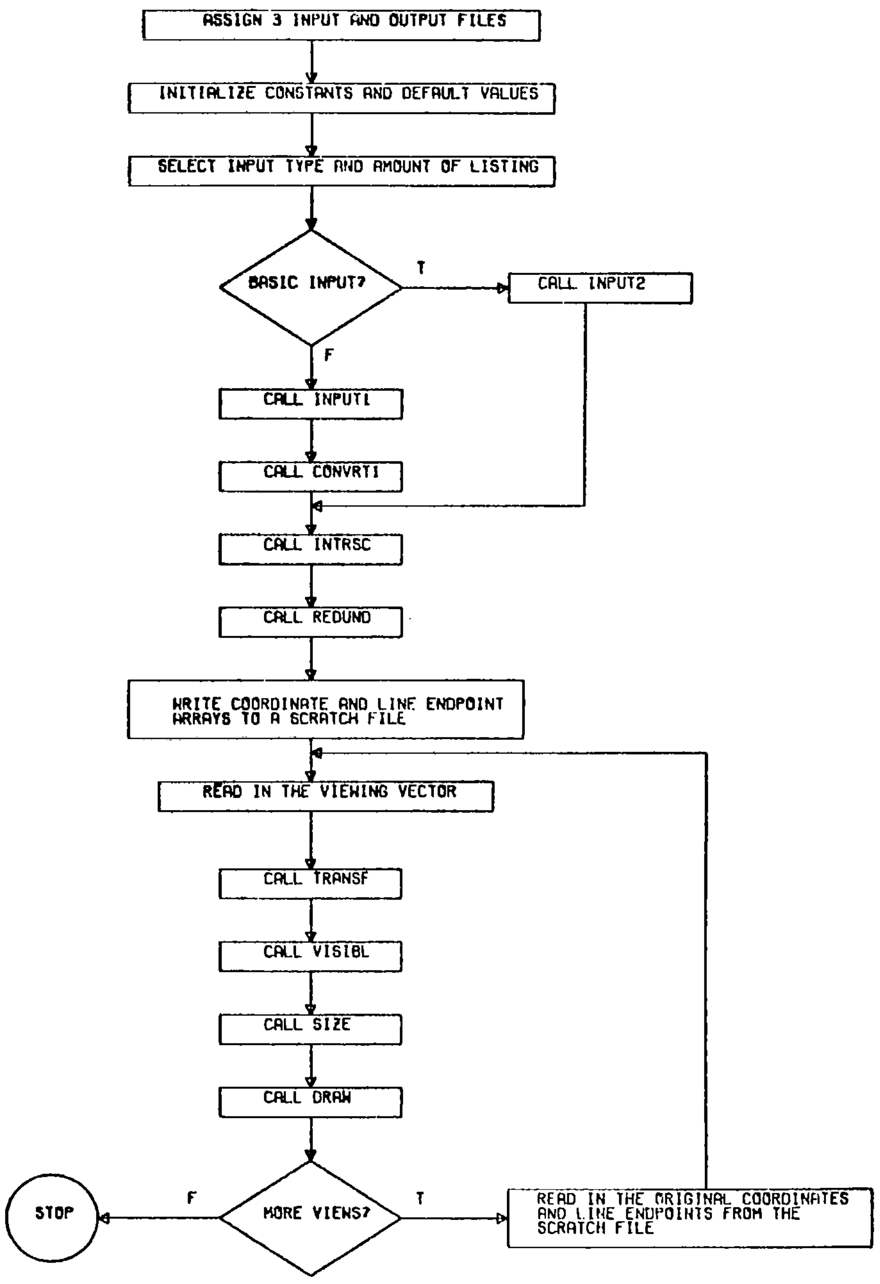

ASSIGN 3 INPUT AND OUTPUT FILES
INITIALIZE CONSTANTS AND DEFAULT VALUES
SELECT INPUT TYPE AND AMOUNT OF LISTING
BASIC INPUT?
T
CALL INPUT2
F
CALL INPUT1
CALL CONVRT1
CALL INTRSC
CALL REDUND
WRITE COORDINATE AND LINE ENDPOINT ARRAYS TO A SCRATCH FILE
READ IN THE VIEWING VECTOR
CALL TRANSF
CALL VISIBL
CALL SIZE
CALL DRAW
MORE VIEWS?
F
STOP
T
READ IN THE ORIGINAL COORDINATES AND LINE ENDPOINTS FROM THE SCRATCH FILE

FIGURE 19. HIDDEN LINE REMOVAL FLOWCHART

Subroutine INPUT2

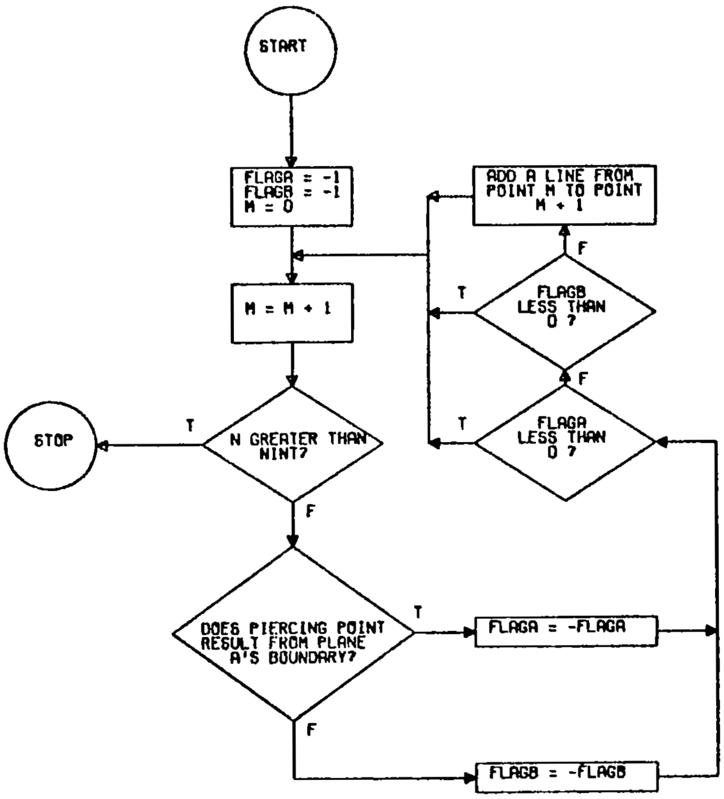
Subroutine INPUT2 contains the code for processing the object description as expressed using the basic input scheme. As the data is being read the coordinate arrays X, Y, and Z are filled as well as the line endpoint arrays LA and LB. The plane and hole vertices are stored on a temporary file. This is done to save core and is feasible because the vertices are only referenced a few times whereas the coordinates and line endpoints are continually being references during program operation. When the routine encounters an "END" keyword or the end of the object input file a return to the main program is executed.

Subroutines INPUT1, COIL, and CNVRT1

INPUT1 is the subroutine which processes objects which have been described using the Composite Object Input Language. This mode of input has not been used in current ODIN applications. The generalized data output arrays are filled in preparation for the CNVRT1 subroutine. Subroutine COIL and the INPUT1 were designed after the Hidden Line Removal Program to provide general three dimensional input and it was felt that the arrays that INPUT1 produces should be more generalized than those required for the Hidden Line Removal Program. Thus, subroutine CNVRT1 is required to convert the data formed by subroutine INPUT1 into the required data. The required data being the file containing the vertices, the coordinate arrays, and the line endpoint arrays.

Subroutine INTRSC - Plane Intersections

Having processed all the object data the program generates all lines of the intersection formed by the intersections of planes. The intersection problem is handled in subroutine INTRSC. The general approach to the problem is to compute all lines of intersection plane 1 forms with plane 2 and subsequent planes, then to compute all lines of intersection plane 2 forms with plane 3 and subsequent planes, and so forth. The addition of lines of intersection to the data simply involves the addition of new points to the coordinate arrays and new sequence numbers to the line endpoint arrays. The points are determined simply by computing all points where the boundaries of each plane pierce the other. If a boundary line does pierce the other plane then it is known that it is one point on a line of intersection. The analytics of a line piercing a plane are quite straight-forward and are discussed in Appendix B. Once all points along the intersection have been determined, the various pairs of points to be connected



FOR THE SAKE OF THIS DIAGRAM CONSIDER THAT THE BOUNDARIES OF TWO PLANES, PLANE A AND PLANE B, HAVE BEEN FOUND TO PIERCE EACH OTHER NINT NUMBER OF TIMES ALONG THE LINE OF INTERSECTION.

FIGURE 20. INTERSECTION LINE CONNECTIVITY LOGIC

becomes the problem. If all the piercing points are placed in a list ordered by their position along the line of intersection then the procedure shown in Figure 20 will generate the appropriate pairs. This logic is adequate enough to figure the connectivity of the lines of intersection which arise from the most complicated intersecting planes.

Subroutine REDUND

The lines which form the boundaries of the planes and holes must also be added to the line tables. This function is performed by subroutine REDUND. The subroutine received its name because of a constraint that is enforced. That constraint is to avoid adding redundant lines. When considering polyhedra it is obvious that most planes share boundaries, therefore, if the boundaries were added indiscriminately the number of lines to be processed and output would be almost doubled what needs to be.

The coordinate and line endpoint arrays are then complete. These arrays will be altered then by transformations and the visibility process so if multiple view generation is to be permitted it is advantageous to save the data at this point. Thus, the coordiante arrays and line endpoint arrays are copied out to a scratch file.

Viewing Vector and Viewing Plane

The viewing vector is read next. This vector is used then to determine the coordinate transformation and perspective projection for the view. The general purpose of the coordinate transformation is to translate the origin of the coordinate system to the point being observed and to rotate the coordinate system so that the positive Z axis contains the observer's position. The analytical method for performing this transformation is given in Appendix C. Once the coordinates have been transformed the perspective projection can be made. This essentially consists of projecting the object points onto a picture plane which is normal to the line of sight vector and passes through the point being observed. The resulting projection is thus two dimensional and lies on the plane $Z = 0$. Two new arrays, XP and YP, are filled with the perspective representation of the coordinates. The actual analytical derivation and equations for the perspective projection are given in Appendix D. With the transformations out of the way the visibility problem is the next one to be solved.

Subroutine VISIBL

The determination of which lines are visible or partially visible is performed by subroutine VISIBL. The general approach taken in this subroutine is to consider each plane and determine which lines the plane hides or partially hides. The lines hidden or partially hidden are removed from the line endpoint arrays and if the line was only partially hidden the remaining visible segment or segments are added to the coordinate and endpoint arrays. After all the planes have been processed only those lines remain in the line endpoint table which should be displayed. Thus, the whole visibility problem reduces to the determination of methodology to resolve where a plane hides a line. The general outline for this methodology as implemented in subroutine VISIBL is shown schematically in Figures 21 and 22.

As can be seen in Figure 21, the VISIBL subroutine begins with the code for referencing each plane. After a plane has been referenced the limiting coordinates are determined and an approximation for 0 is formed based on a percentage of the size of the plane. This approximation to 0 is required for subsequent calculations where round-off is a problem. The constants for the plane equation, $AX + BY + CZ + D = 0$, are then computed. Based on these constants the planes which appear as edge views may be trivially rejected because a plane which appears as an edge will not hide any lines. Next those planes which are part of convex objects and have had their vertices entered so as to generate an outer normal are considered. If the outer normal has a negative Z component the C in plane equation is negative and the whole plane need not be considered. This is so because the negative C means the plane is on the back of the convex polyhedron and anything the plane would hide will be hidden by the planes on the front of the polyhedron. There is no way to indicate a convex as opposed to a concave solid using the basic input scheme so this feature is only accessible through inputs such as COIL where the computer handles the computation of the direction of vertex input.

Next the line endpoint arrays are compressed or have those lines eliminated which were indicated to be removed when the last plane was processed. The current number of lines is then stored because as the current plane is processed the total number of lines grwos with the generation of the line segments not hidden. There is then no need to consider these new lines against the plane when it is known they are not hidden by the current plane. Thus, by storing the number of lines in the table when the consideration of the plane begins it is knwon how many lines to consider even though the total number of lines increases. With the number of lines at the start stored, each line may then be considered.

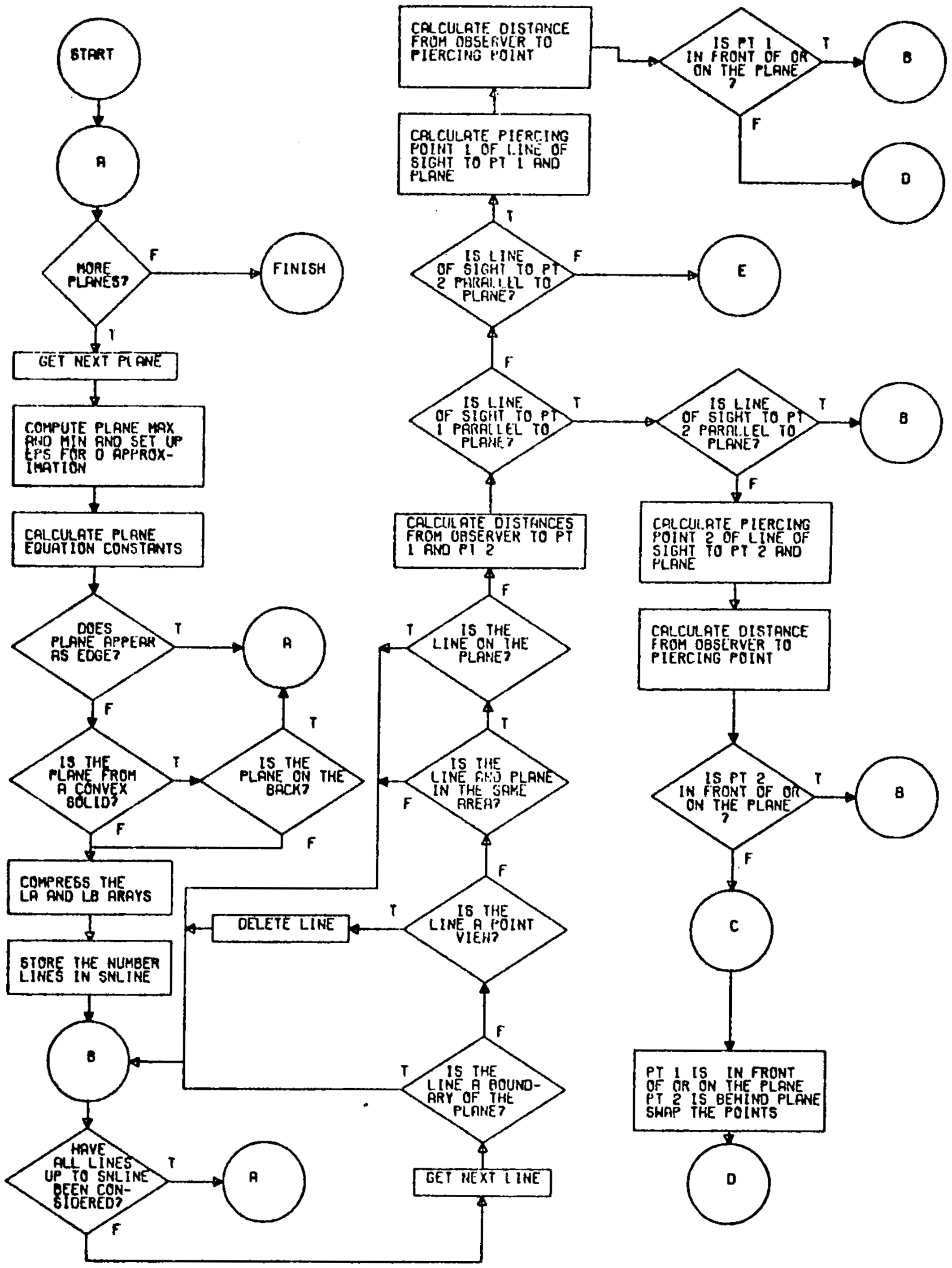

START
A
MORE PLANES?
F
FINISH
T
GET NEXT PLANE
COMPUTE PLANE MAX AND MIN AND SET UP EPS FOR 0 APPROX-IMATION
CALCULATE PLANE EQUATION CONSTANTS
DOES PLANE APPEAR AS EDGE?
T
A
F
IS THE PLANE FROM A CONVEX SOLID?
T
IS THE PLANE ON THE BACK?
T
F
F
COMPRESS THE LA AND LB ARAYS
STORE THE NUMBER LINES IN SNLINE
B
HAVE ALL LINES UP TO SNLINE BEEN CON-SIDERED?
T
A
F
GET NEXT LINE
IS THE LINE A BOUND-ARY OF THE PLANE?
T
F
IS THE LINE A POINT VIEW?
T
DELETE LINE
F
IS THE LINE AND PLANE IN THE SAME AREA?
F
T
IS THE LINE ON THE PLANE?
T
F
CALCULATE DISTANCES FROM OBSERVER TO PT 1 AND PT 2
IS LINE OF SIGHT TO PT 1 PARALLEL TO PLANE?
T
F
IS LINE OF SIGHT TO PT 2 PARALLEL TO PLANE?
T
B
F
IS LINE OF SIGHT TO PT 2 PARALLEL TO PLANE?
F
E
T
CALCULATE PIERCING POINT 1 OF LINE OF SIGHT TO PT 1 AND PLANE
CALCULATE DISTANCE FROM OBSERVER TO PIERCING POINT
IS PT 1 IN FRONT OF OR ON THE PLANE ?
T
B
F
D
CALCULATE PIERCING POINT 2 OF LINE OF SIGHT TO PT 2 AND PLANE
CALCULATE DISTANCE FROM OBSERVER TO PIERCING POINT
IS PT 2 IN FRONT OF OR ON THE PLANE ?
T
B
F
C
PT 1 IS IN FRONT OF OR ON THE PLANE PT 2 IS BEHIND PLANE SWAP THE POINTS
D

FIGURE 21. SUBROUTINE VISIBL FLOWCHART

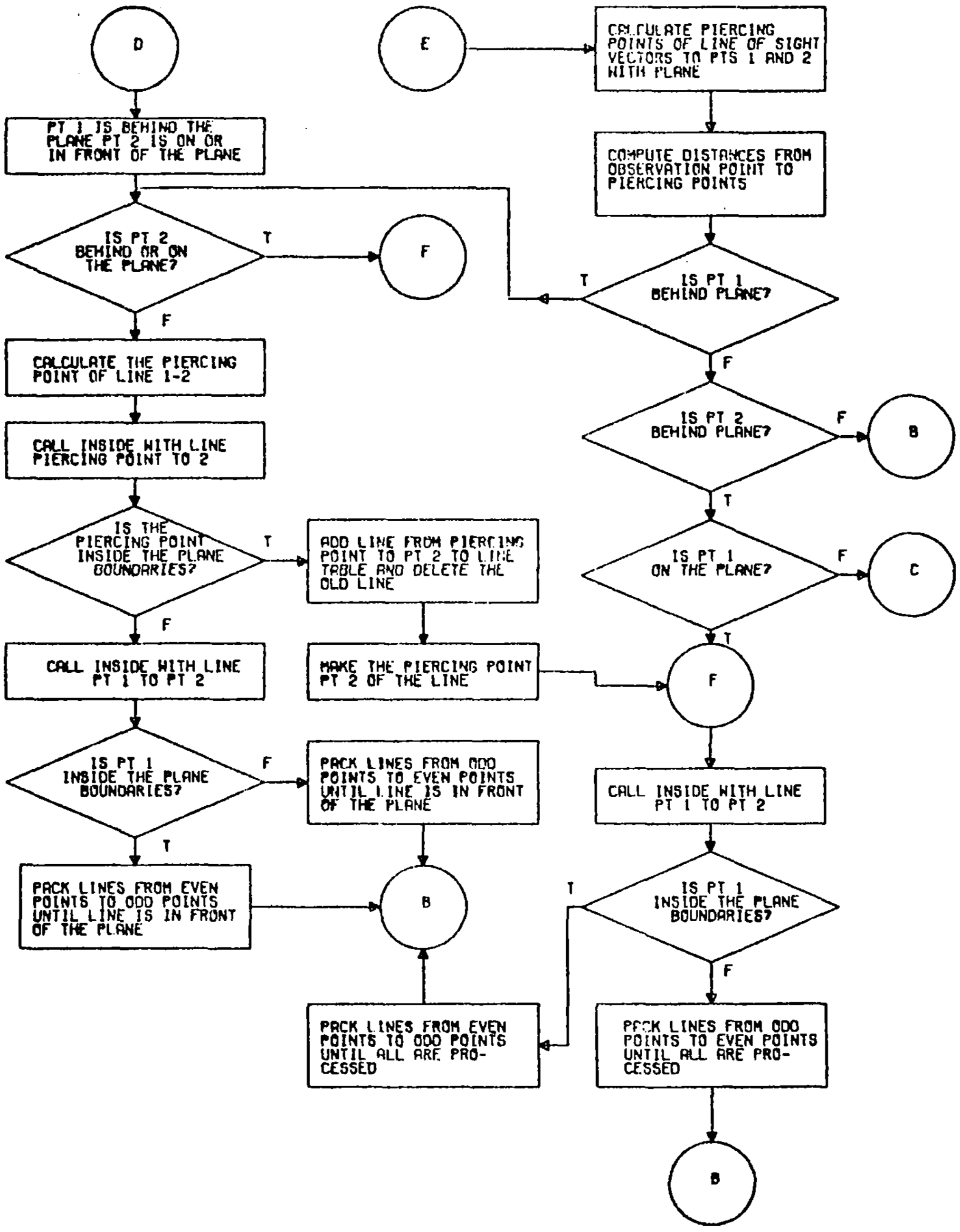

D
E
CALCULATE PIERCING POINTS OF LINE OF SIGHT VECTORS TO PTS 1 AND 2 WITH PLANE
PT 1 IS BEHIND THE PLANE PT 2 IS ON OR IN FRONT OF THE PLANE
COMPUTE DISTANCES FROM OBSERVATION POINT TO PIERCING POINTS
IS PT 2 BEHIND OR ON THE PLANE?
T
F
F
IS PT 1 BEHIND PLANE?
T
F
CALCULATE THE PIERCING POINT OF LINE 1-2
IS PT 2 BEHIND PLANE?
F
B
T
CALL INSIDE WITH LINE PIERCING POINT TO 2
IS THE PIERCING POINT INSIDE THE PLANE BOUNDARIES?
T
ADD LINE FROM PIERCING POINT TO PT 2 TO LINE TABLE AND DELETE THE OLD LINE
IS PT 1 ON THE PLANE?
F
C
F
T
CALL INSIDE WITH LINE PT 1 TO PT 2
MAKE THE PIERCING POINT PT 2 OF THE LINE
F
IS PT 1 INSIDE THE PLANE BOUNDARIES?
F
PACK LINES FROM ODD POINTS TO EVEN POINTS UNTIL LINE IS IN FRONT OF THE PLANE
CALL INSIDE WITH LINE PT 1 TO PT 2
T
PACK LINES FROM EVEN POINTS TO ODD POINTS UNTIL LINE IS IN FRONT OF THE PLANE
B
T
IS PT 1 INSIDE THE PLANE BOUNDARIES?
F
PACK LINES FROM EVEN POINTS TO ODD POINTS UNTIL ALL ARE PROCESSED
PACK LINES FROM ODD POINTS TO EVEN POINTS UNTIL ALL ARE PROCESSED
B

FIGURE 22. SUBROUTINE VISIBLE FLOW CHART (PART 2)

There are several trivial cases which can immediately and easily be resolved. First, if the line is a boundary of the plane it cannot be hidden by the plane so the next line may be considered. If the line is a point view it need not be considered further and may be removed from the line table. Using the maximum and minimum plane coordinates determined earlier it is also possible to avoid further consideration of lines which do not even lie in the same reion as the plane. If a line is still to be considered after all of these constraints, more involved computations are required.

The initial computations are involved in determining if either endpoint is behind the plane at least in depth. This first level of consideration ignores whether the line lies outside or inside the boundaries of the plane as projected on the picture plane and considers the problem as if a line and infinite plane were the case. The depth consideration is based on the distances from the observer to the line endpoints and to the projections of the line endpoints and to the projections of the line endpoints onto the plane. Thus, the first calculations compute the distances from the observer to line endpoints 1 and 2. Then it is attempted to project the points back to the plane. If the line through the observation point and line endpoint could not be projected onto the plane and is on the visible side of the plane. If the projection line is not parallel to the plane then the piercing point of the projection line and plane is computed along with the distance from the observer to this piercing point. If both line endpoints are in front of or on the plane, the plane cannot hide the line and the next line may be considered. If the line intersects the plane or lies entirely behind the plane, additional processing is necessary.

If the line intersects the plane, point 1 will be behind the plane and point 2 will be in front of the plane. The piercing point is computed and then it is needed to know whether the piercing point occurs inside or outside the plane's boundaries. A piercing point which occurs within a hole or notch in the plane must obviously be considered outside the plane.

Subroutine INSIDE

Subroutine INSIDE decides whether the first point of the line supplied to it is inside or outside the plane's boundaries as well as returning all apparent intersection points between the line and plane boundaries as projected in the picture plane. INSIDE also indicates whether the line is in front of or behind the plane at each intersection point. If the piercing point falls within the plane boundaries, the original line is deleted and a line from the peircing point to endpoint 2 is added to the line endpoint arrays with the piercing point being added to the coordinate arrays. Point 2 of the original line is replaced by the piercing point coordinates and the modified line is then considered as a line lying totally behind the plane as is described in the next paragraph. If, however, the piercing point falls outside the boundaries of the plane the line is sent to INSIDE to determine if point 1 is inside or outside the boundaries of the plane.

Again as well as resolving the intersection problem all apparent intersections of the line and plane boundaries along with depth information are returned. When the list of apparent intersection points is returned line endpoint 1 is the first in the list, line endpoint 2 is the last in the list and all others are appropriately ordered in between. If point 1 was inside the plane's boundaries, lines are added from the even apparent intersections to the odd apparent intersections until the apparent intersections are in front of the plane. If point 1 was outside the plane's boundaries, lines are added from the odd apparent intersections to the even apparent intersections until the apparent intersections are in front of the plane. In both cases when an apparent intersection was finally found which was in front of the plane a line was added from the previous apparent intersection to endpoint 2.

When the line lies entirely behind the plane the procedure is greatly simplified over that required when the line intersects teh plane. The line which lies entirely behind the plane is first sent to INSIDE to find out if point 1 is inside or outside the plane's boundaries and to determine the apparent intersections. If point 1 is inside the plane, lines are added from even apparent intersection points to odd apparent intersection points until all the apparent intersections have been considered. However, if point 1 is outside the plane, lines are added from odd apparent intersection points to even apparent intersection points until all the apparent intersections have been considered.

Whenever a newly generated line segment is added the original line from which the line segment was generated is deleted.

The logic to decide whether the first point of a line is inside or outside the plane is much simpler than might originally be anticipated. The line is extended from point 1 through point 2 until the new endpoint is beyond the plane. The maxima and minima computed earlier for the plane are employed for this. Then all apparent intersections are computed for the extended line and the boundary lines for the plane as they appear on the picture plane. Any hole boundary lines are also considered at this point because they too are boundaries of the plane. The apparent intersectionsare projected back to the plane and line and the distances are computed from the observer to the points on the plane and line respectively. Then if the number of apparent intersections is odd the first point is inside the plane and if it is even the first point is outside the plane.

Subroutines SIZE and DRAW

After VISIBL has produced line endpoint arrays with only the lines to be displayed, subroutine SIZE can use this data to compute the maxima and minima picture plane coordinates of the points to be displayed. The maxima and minima can then be used along with the plot size to compute a scale factor. The minima and scale factor are then passed through common and argument list to subroutine DRAW. Subroutine DRAW writes out the X minimum, Y minimum, and scale factor in a 2H 1,3F7.2 field and then outputs the lines. The lines are output by using the sequence numbers from the line endpoint arrays to reference the pairs of coordinates from the perspective coordinate arrays. The pairs of coordinates are then simply written out in a 2X,4F.7.2 format.

Having written out the information for the view, the original coordinate and line data is retrieved from the scratch file and the next view is considered. Upon completion of all views the program execution is terminated.

The output file which was created can then be used as input by programs which drive the appropriate graphics device, for example, program PLOTTR, reference 7. These programs read teh scaling and minima information and use it in the plotting of the lines.

APPENDIX B - PART II

COMPUTATION OF THE PIERCING POINT OF A LINE THROUGH A PLANE

The computation of the piercing point of a line through a plane is nothing more than the simultaneous solution of the equations for the line and a plane. The equation of a plane is given by

$$Ax + By + Cz + D = 0 \tag{1}$$

and that of a line by

$$\frac{x-X_1}{X_2-X_1} = \frac{y-Y_1}{Y_2-Y_1} = \frac{z-Z_1}{Z_2-Z_1} \tag{2}$$

where X_1, Y_1, Z_1 and X_2, Y_2, Z_2 are the line endpoints.

Solving equation 2 for y and z in terms of x yields

$$y = (x-X_1)\,\frac{Y_2-Y_1}{X_2-X_1} + Y_1 \tag{3}$$

$$z = (x-X_1)\,\frac{Z_2-Z_1}{X_2-Z_1} + Z_1 \tag{4}$$

Substituting y and z as given in equations 3 and 4 into equation 1 yields

$$Ax + B[(x-X_1)\,(\frac{Y_2-Y_1}{X_2-X_1}) + Y_1] + C[(x-X_1)\,(\frac{z-Z_1}{X_2-X_1}) + Z_1]$$
$$+ D = 0$$

or

$$x[A + B(\frac{Y_2-Y_1}{X_2-X_1}) + C(\frac{Z_2-Z_1}{X_2-X_1})] = BX_1(\frac{Y_2-Y_1}{X_2-X_1}) - BY_1 +$$
$$CX_1(\frac{Z_2-Z_1}{X_2-X_1}) - CZ_1 - D\ .$$

Solving for x,

$$x = \frac{B[X_1(\frac{Y_2-Y_1}{X_2-X_1}) - Y_1] + C[X_1(\frac{Z_2-Z_1}{X_2-X_1}) - Z_1] - D}{A + B(\frac{Y_2-Y_1}{X_2-X_1}) + C(\frac{Z_2-Z_1}{X_2-X_1})} .$$

Then letting

$$X_2-X_1 = V, \qquad (5)$$

$$\frac{Y_2-Y_1}{X_2-X_1} = \frac{Y_2-Y_1}{V} = U, \qquad (6)$$

$$\frac{Z_2-Z_1}{X_2-X_1} = \frac{Z_2-Z_1}{V} = W, \qquad (7)$$

the equation for x becomes

$$x = \frac{B(X_1U-Y_1) + C(X_1W-Z_1) - D}{A + BU + CW} . \qquad (8)$$

Equations 3 and 4 may then be used to compute the y and z coordinate of the piercing point as

$$y = (x-X_1)\ U + Y_1 \qquad (9)$$

and

$$z = (x-X_1)\ W + Z_1 \qquad (10)$$

If V = 0, equations 6 and 7 yield ∞. So the line equations are solved for x and z in terms of y and these equations are substituted back into equation 1. Equation 1 is solved for y and with

$$V = Y_2-Y_1$$

$$U = \frac{X_2-X_1}{Y_2-Y_1} = \frac{X_2-X_1}{V} = 0$$

$$W = \frac{Z_2-Z_1}{Y_2-Y_1} = \frac{Z_2-Z_1}{V}$$

yields

$$y = \frac{-AX_1 + C[Y_1W-Z_1] - D}{B + CW}$$

$$x = X_1 \text{ or } X_2$$

$$z = (y-Y_1)\,W + Z_1$$

If both $X_2-X_1 = 0$ and $Y_2-Y_1 = 0$

$$z = \frac{-AX_1 - BY_1 - D}{C}$$

and

$$x = X_1 \text{ or } X_2$$

$$y = Y_1 \text{ or } Y_2 \ .$$

APPENDIX C - PART II

COORDINATE TRANSFORMATION

In order to create the various views desired, the coordinates of an object must be transformed. The transformation consists of changing the coordinate system from its original orientation to an orientation in which the origin is the point at which the observer is looking, (XL,YL,ZL), the positive Z axis is aimed toward the point where the observer is located, (X0,Y0,Z0). This transformation is thus a translation and two rotations as shown in Figure C-1.

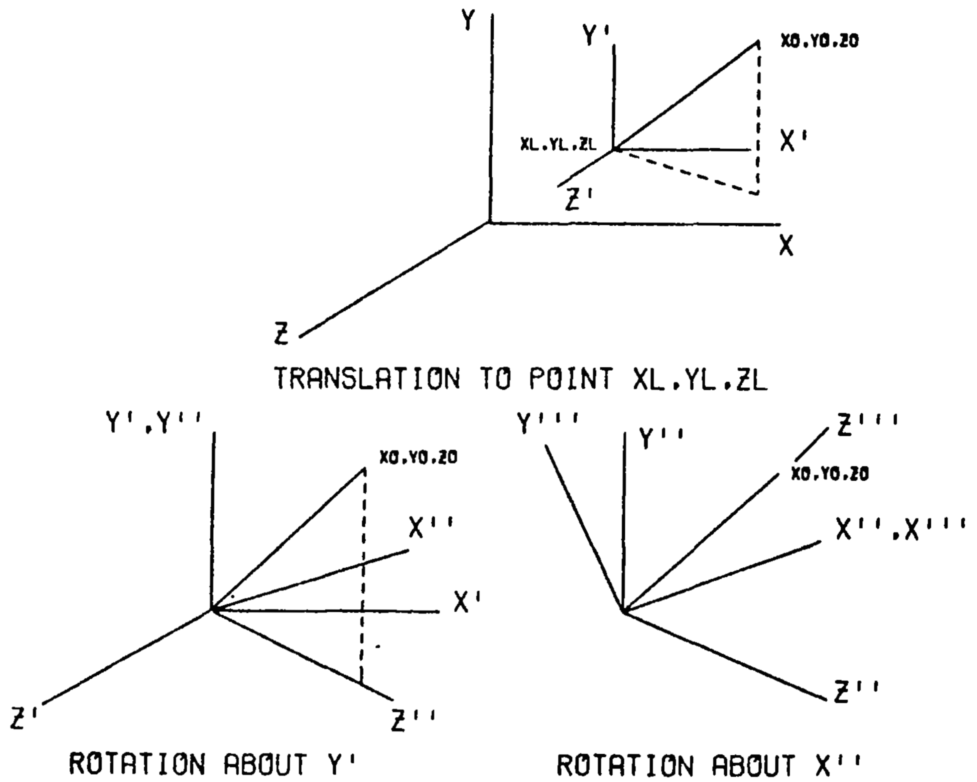


FIGURE C-1. COORDINATE TRANSFORMATION

The transformations to perform each of these tasks will be defined and then concatenated.

The translation is the simplest transformation and can be expressed as

$$X-XL = X'$$

$$Y-YL = Y'$$

$$Z-ZL = Z'$$

in matrix form with homogeneous coordinates the transformation becomes

$$\begin{bmatrix} 1 & 0 & 0 & -XL \\ 0 & 1 & 0 & -YL \\ 0 & 0 & 1 & -ZL \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} X' \\ Y' \\ Z' \\ 1 \end{bmatrix}$$

or

$$[T_T]\ [R] = [R']$$

Two rotations are required as shown in Figure C-1. The first is a rotation about Y' to bring Z' under the observation vector and the second is a rotation about X" to make Z" and the observation vector coicident.

The problem is to determine the new coordinates (R') in the rotated coordinate system from the coordinates (R) in the old system. The general transformation, (Hall, 1966), is given below for the rotation of the coordinate system about an arbitrary axis.

$$\begin{bmatrix} (U_X{}^2 \text{vers}\gamma+\cos\gamma) & (U_X U_Y \text{vers}\gamma+U_Z \sin\gamma) & (U_X U_Z \text{vers}\gamma-U_Y \sin\gamma) \\ (U_X U_Y \text{vers}\gamma-U_Z \sin\gamma) & (U^2{}_Y \text{vers}\gamma+\cos\gamma) & (U_Y U_Z \text{vers}\gamma+U_X \sin\gamma) \\ (U_Z U_X \text{vers}\gamma+U_Y \sin\gamma) & (U_Y U_Z \text{vers}\gamma-U_X \sin\gamma) & (U_Z{}^2 \text{vers}\gamma+\cos\gamma) \end{bmatrix} = [T]$$

$$[T]\ [R] = [R''']$$

where U is unit vector along the axis of rotation, γ the angle of rotation, and vers = 1-cosγ.

If the transformation and vectors are extended to homogeneous coordinates:

$$\begin{bmatrix} (U_X{}^2\text{vers}\gamma+\cos\gamma) & (U_XU_Y\text{vers}\gamma+U_Z\sin\gamma) & (U_XU_Z\text{vers}\gamma-U_Y\sin\gamma) & 0 \\ (U_XU_Y\text{vers}\gamma-U_Z\sin\gamma) & (U_Y{}^2\text{vers}\gamma+\cos\gamma) & (U_YU_Z\text{vers}\gamma+U_X\sin\gamma) & 0 \\ (U_ZU_X\text{vers}\gamma+U_Y\sin\gamma) & (U_YU_Z\text{vers}\gamma-U_X\sin\gamma) & (U_Z{}^2\text{vers}\gamma+\cos\gamma) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = [T]$$

$$[R] = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \qquad [R'] = \begin{bmatrix} X' \\ Y' \\ Z' \\ 1 \end{bmatrix}$$

Now if a rotation about Y is desired with an angle of rotation α. Then $U_X=U_Z=0$ and $U_Y=1$, $\gamma=\alpha$, the transformation (T) becomes:

$$\begin{bmatrix} \cos\alpha & 0 & -\sin\alpha & 0 \\ 0 & 1 & 0 & 0 \\ \sin\alpha & 0 & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = [T_Y]$$

If a rotation about X is desired with an angle of rotation β. Then $U_Y=U_Z$ 0, $U_X=1$, $\gamma=\beta$, and the transformation (T) becomes;

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\beta & \sin\beta & 0 \\ 0 & -\sin\beta & \cos\beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = [T_X]$$

Thus the total transformation can now be written as

$$[T_X]\ [T_Y]\ [T_T]\ [R] = [R''']$$

Expanding (T_T) (R) =

$$\begin{bmatrix} X-XL \\ Y-YL \\ Z-ZL \\ 1 \end{bmatrix}$$

Multiply that result by (T_Y)

$$\begin{bmatrix} \cos\alpha(X-XL) - (Z-ZL)\sin\alpha \\ Y-YL \\ (X-XL)\sin\alpha + (Z-ZL)\cos\alpha \\ 1 \end{bmatrix}$$

Multiplying that result by (T_X)

$$\begin{bmatrix} (X-XL)\cos\alpha-(Z-ZL)\sin\alpha \\ (Y-YL)\cos\beta+(X-XL)\sin\alpha\sin\beta+(Z-ZL)\cos\alpha\sin\beta \\ -(Y-YL)\sin\beta+(X-XL)\sin\alpha\cos\beta+(Z-ZL)\cos\alpha\cos\beta \\ 1 \end{bmatrix} = \begin{bmatrix} X''' \\ Y''' \\ Z''' \\ 1 \end{bmatrix}$$

APPENDIX D - PART II

PERSPECTIVE TRANSFORMATION

Once the Z axis is aligned with the line of sight vector the points must then be projected onto the YX plane to generate teh desired perspective view. This projection is shown in Figure D-1.

The analytical derivation is quite straight-forward. Noting similar triangles in Figure D-1 the following equations may be written

$$\frac{YP}{DIST} = \frac{Y}{DIST-Z}$$

and

$$\frac{XP}{DIST} = \frac{X}{DIST-Z}$$

Thus

$$XP = \frac{XDIST}{DIST-Z}$$

and

$$YP = \frac{YDIST}{DIST-Z}$$

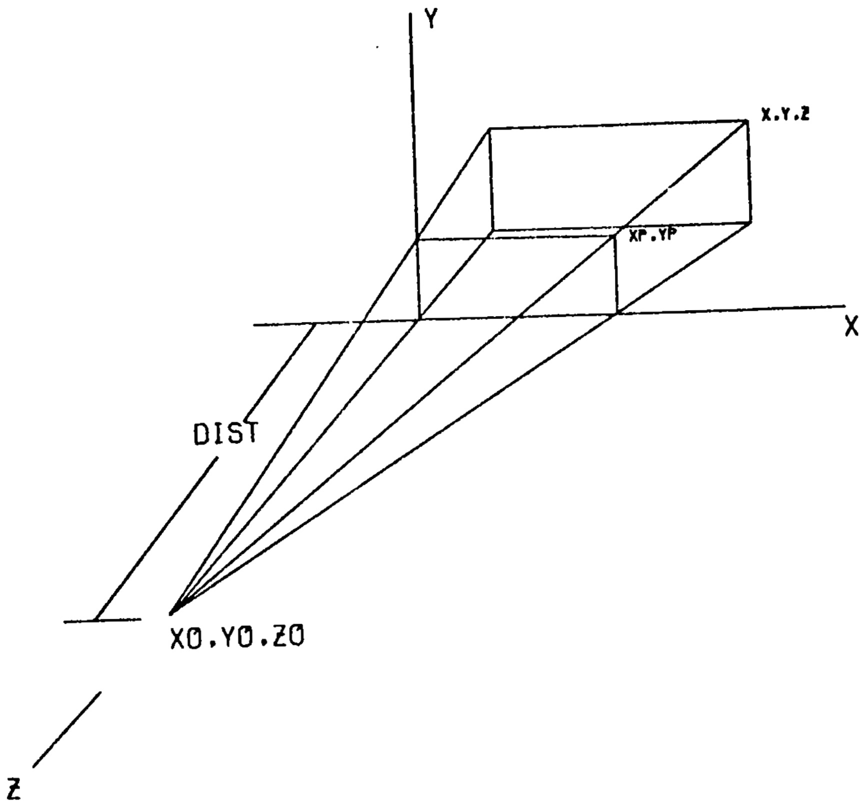


FIGURE D-1. PERSPECTIVE TRANSFORMATION

PART III - INTERFACING PROGRAMS IMAGE AND HIDDEN

Aerophysics Research Corporation has constructed a program interface between the IMAGE and HIDDEN codes. This interface permits program HIDDEN to operate on program IMAGE data. Thus, when the ODIN system creates an input file for IMAGE the same file may be used to draw accurate hidden line pictures through HIDDEN. The two picture drawing codes are therefore available to ODIN system users with a unified input procedure. In view of the time consuming calculations required when producing true hidden line pictures through program HIDDEN a user has the option of "thinning out" the IMAGE data if desired.

IMAGE SUMMARY

Image Input

IMAGE input consists of corner points in the form

$$\boxed{X_1,\ Y_1,\ Z_1,\ I_1,\ X_2,\ Y_2,\ Z_2,\ I_2}$$

where X is a coordinate normal to the paper plane. The hidden line algorithm uses Z as the normal to the paper plane, hence, CVRTHD assumes the data (on UNIT 5) has the form

$$\boxed{Z_1,\ X_1,\ Y_1,\ I_1,\ Z_2,\ X_2,\ Y_2,\ I_2}$$

The integer I_1 determines corner point status as follows:

I_1 = 2, Start of new vehicle section

= 1, Start of new row

= 0, Internal point

An end-of-file test is used to define the end of data. It is assumed that the vehicle geometry is given only on one side of the plane of symetry.

Program CVRTHD - Data Conversion Program

Main program for converting IMAGE data to HIDDEN data. It uses two subroutines, PNTS and PLANE. The subroutine PNTS first converts the IMAGE corner points into HIDDEN format. Subroutine PLANE connects these points together to supply the "PLANES" data required by HIDDEN. A flow chart for data conversion is given in Figure 23. Data input to CVRTHD is:

First Card (Format I2)

IDIV = Averaging factor for reduction in number of points. End points are fixed but internal points are averaged by combining IDIV cards. Where the number of cards in a row leaves a remainder less than IDIV the program averages as many as possible . Some typical examples of this averaging process are shown in Figure 24.

Remaining Cards (Format 3(F9.3, IX), 1, 3(F9.3, IX), (I1)

The variables

$$\boxed{Z_1, X_1, Y_1, I_1, Z_2, X_2, Y_2, I_2}$$

which successively define the panelled surface.

Subroutine PNTS reproduces them in the form

$$\boxed{\begin{array}{l} Z_1, X_1, Y_1, \\ Z_2, X_2, Y_2, \\ Z_3, X_3, Y_3, \\ \quad\;\vdots \\ Z_n, X_n, Y_n \end{array}}$$

Subroutine PLANE then forms the triangular or quadrilateral panels using Z_i, X_i, Y_i; i = 1, 2,....N as illustrated in Figure 25. Planes are output as integers, for example.

```
1  3  2  1
1  4  3  1
1  5  4  1
1  6  5  1
1  7  6  1
1  8  7  1
1  9  8  1
2  3 11 10  2
3  4 12 11  3
      |
      |
      |
     etc.
```

where these integers correspond to those on Figure 25b.Both points and planes data are output on file "CVTOUT." In Figure 25(a) a typical panelling is illustrated with all points distinct. Figure 25(b) illustrates a case where one row of points are all equal. For example at a fuselage nose. A series of triangular panels are required in HIDDEN at the aircraft nose for zero length lines are not acceptable to

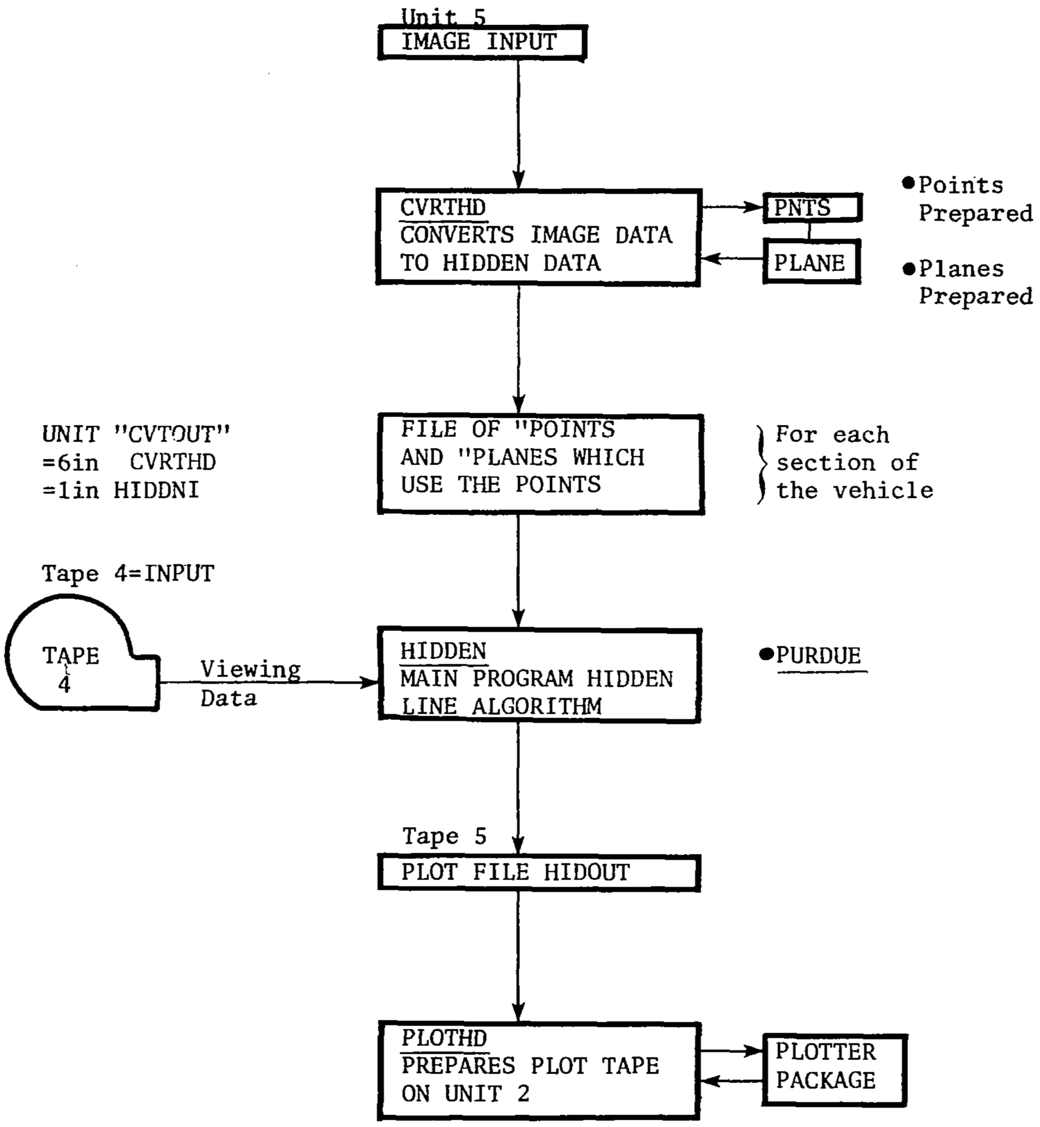
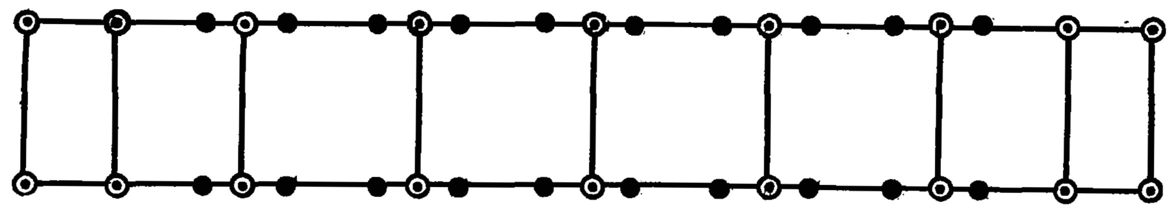
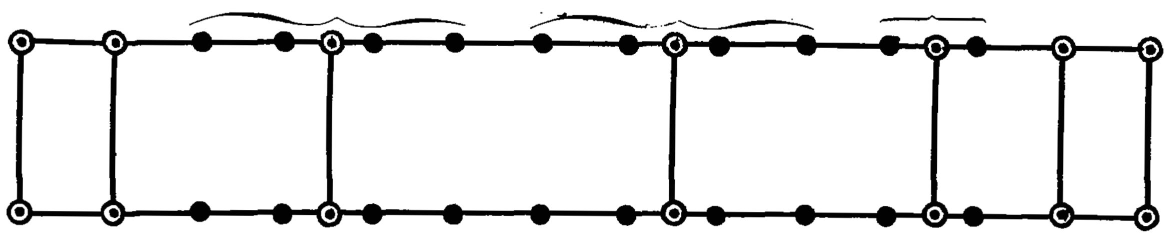


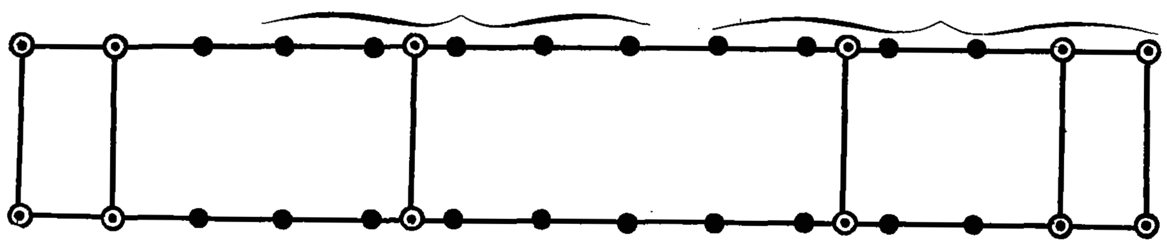
FIGURE 23. SCHEMATIC OF HIDDEN LINE OPTION



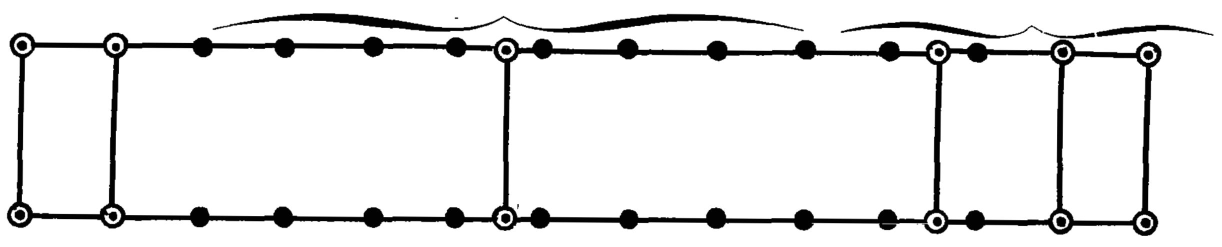
IDIV=1



IDIV=2



IDIV=3



IDIV=4

FIGURE 24. TYPICAL POINT AVERAGING

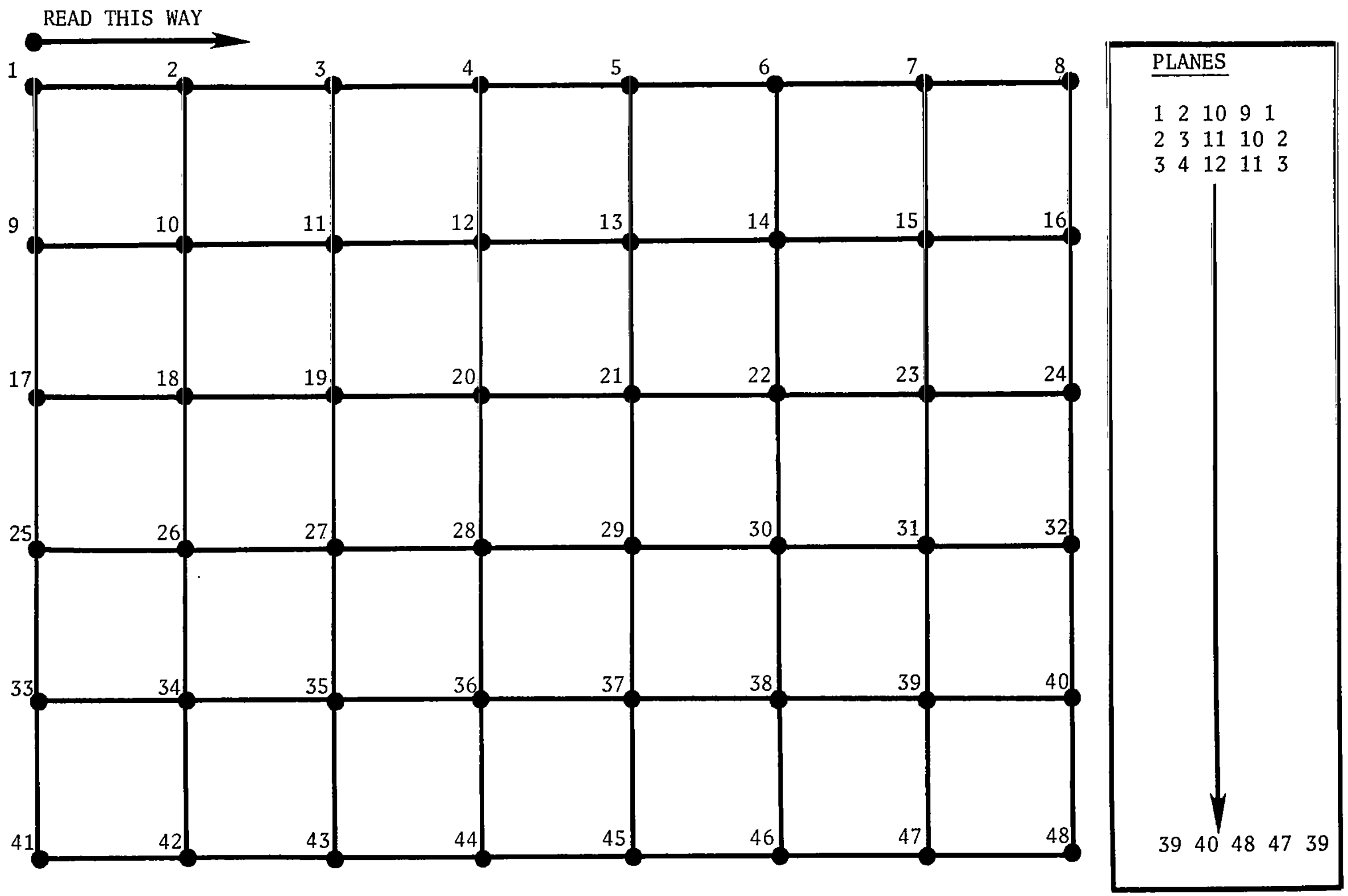
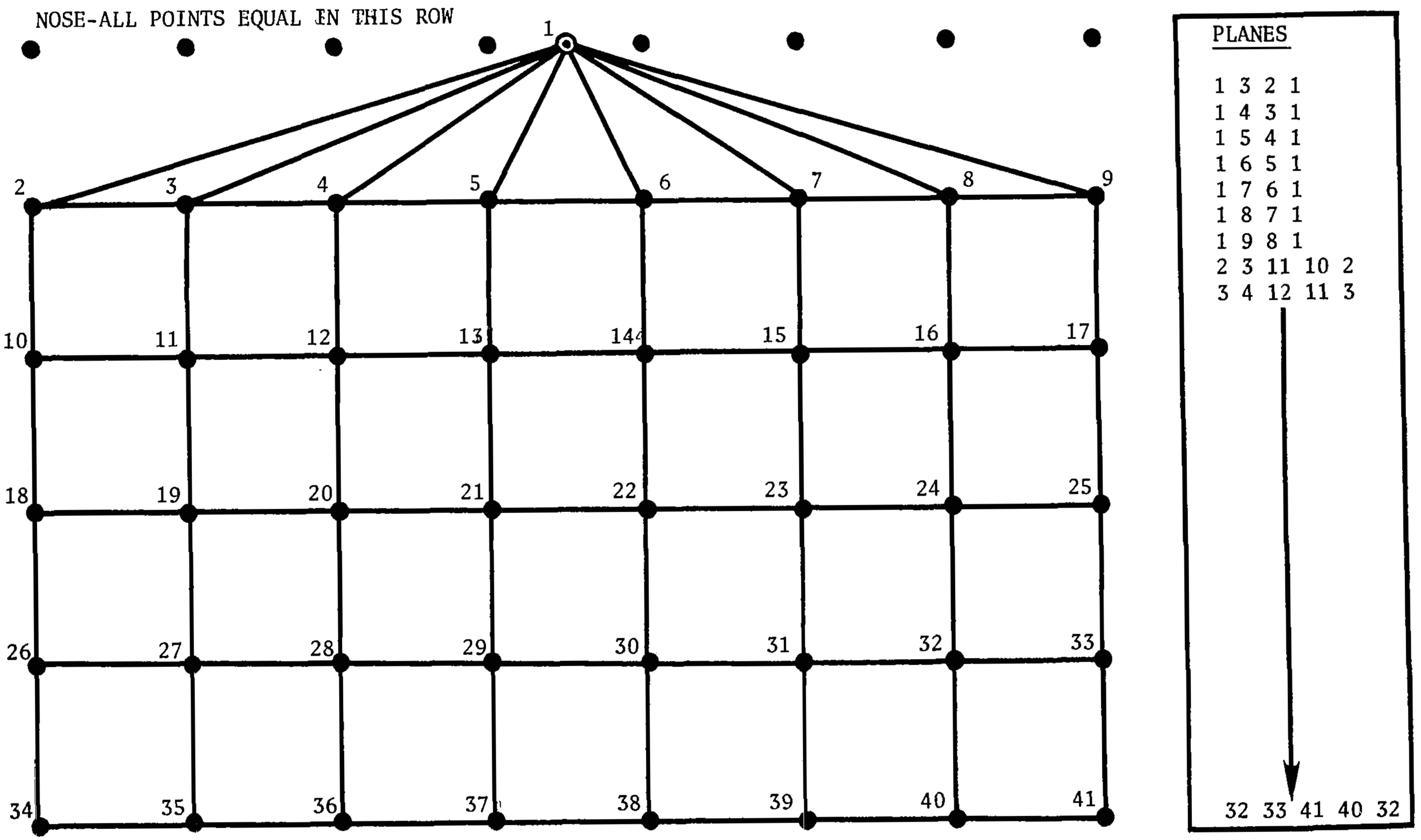


FIGURE 25(a). PLANE CONVENTION ALL POINTS DISTINCT



PLANES

1 3 2 1
1 4 3 1
1 5 4 1
1 6 5 1
1 7 6 1
1 8 7 1
1 9 8 1
2 3 11 10 2
3 4 12 11 3

32 33 41 40 32

FIGURE 25(b). PLANE CONVENTION SEVERAL POINTS COINCIDENT

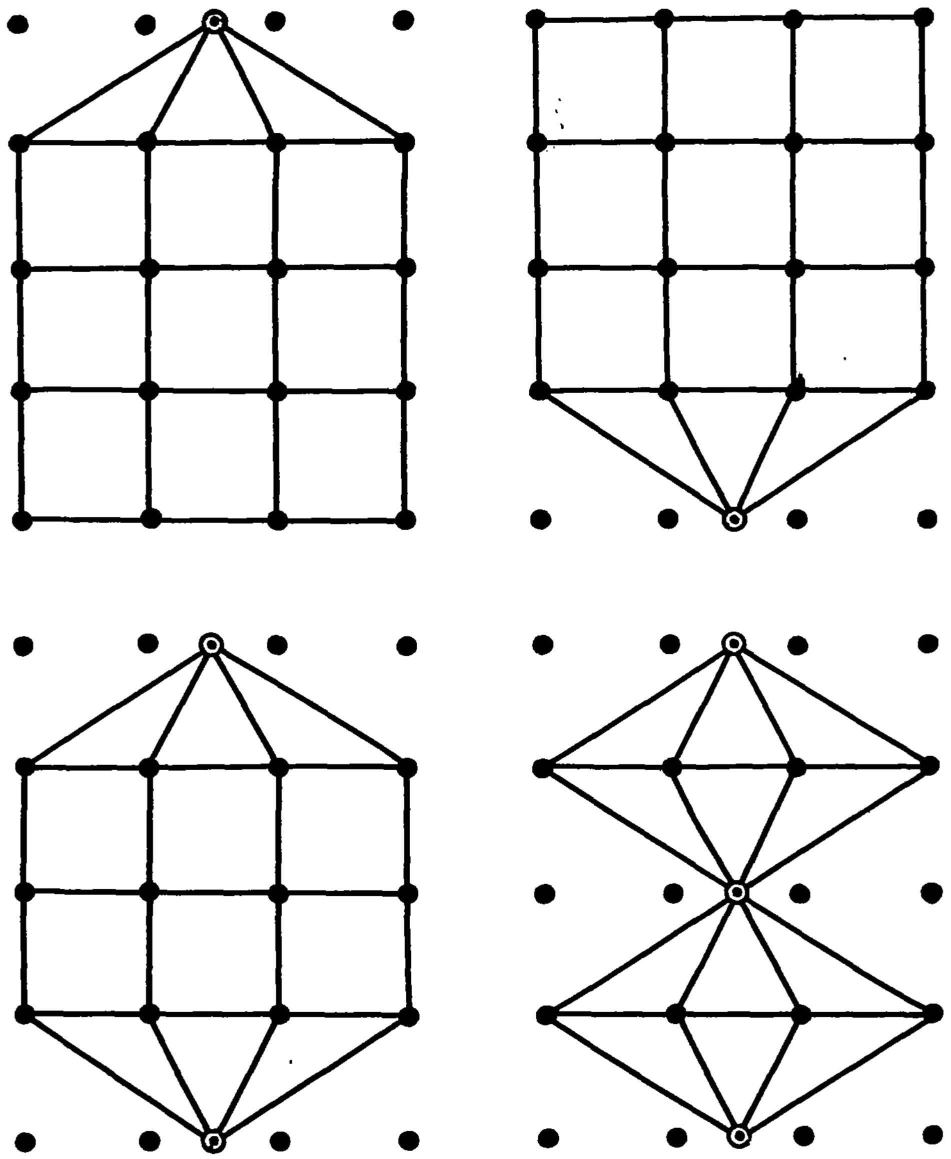


FIGURE 26. TYPICAL PLANAR ELEMENT CONVENTIONS POSSIBLE IN CVRTHD

HIDDEN. Figure 26 illustrates other simple panellings that may be encountered. Program CVRTHD automatically redefines quadrilateral panels as triangular panels wherever this is necessary.

Program HIDDEN Input and Output

Input

The input to HIDDEN consists of points and planes on file CVTOUT as discussed previously. File CVTOUT ends with the keyword END starting in column 1. The format of this file is therefore

```
POINTS
Point triplets for 1st section
PLANES
Plane integers for 1st section
POINTS
Point triplets for 2nd section
PLANES
Plane integers for 2nd section
            |
            |
            |
           etc.
POINTS
Points triplets for last section
PLANES
Plane integers for last section
END
```

The second input file to HIDDEN is the last card of the input data deck. This card has the format 6F10.0. The six quantities on this card are

$$\boxed{XV_1, YV_1, ZV_1, XV_2, YV_2, ZV_2}$$

where these two triplets define the viewing angle. The point XV_1, YV_1, AV_1 should be placed near the vehicle. The point

XV_2, YV_2, ZV_2 should be placed at the viewers location.

Data deck input for a hidden line run therefore has the form illustrated in Figure 27. The viewing vector card is internally transferred to Unit 4 in HIDDEN by file substitution.

Output

Program HIDDEN outputs the file HIDOUT on Tape 5. This file is then read by program PLOTHD in preparing the plot file. The contents of file HIDOUT are in the form:

Record 1 - Scaling Data

$\boxed{\text{J, XOFFSET, YOFFSET, SCALE}}$

where,

J = Record status flag

= 1, First record-scaling data

= 0, Any other record

XOFFSET	=	Internally computed offset for X
YOFFSET	=	Internally computed offset for Y
SCALE	=	Internal scale factor for adjusting picture size in HIDDEN (picture size can be modified in PLOTHD)

Succeeding Records - Line Segments

$\boxed{J,\ X_1,\ Y_1,\ X_2,\ Y_2}$

where,

J = 0

X_1, Y_1 are 1st endpoint of line segment

X_2, Y_2 are 2nd endpoint of line segment

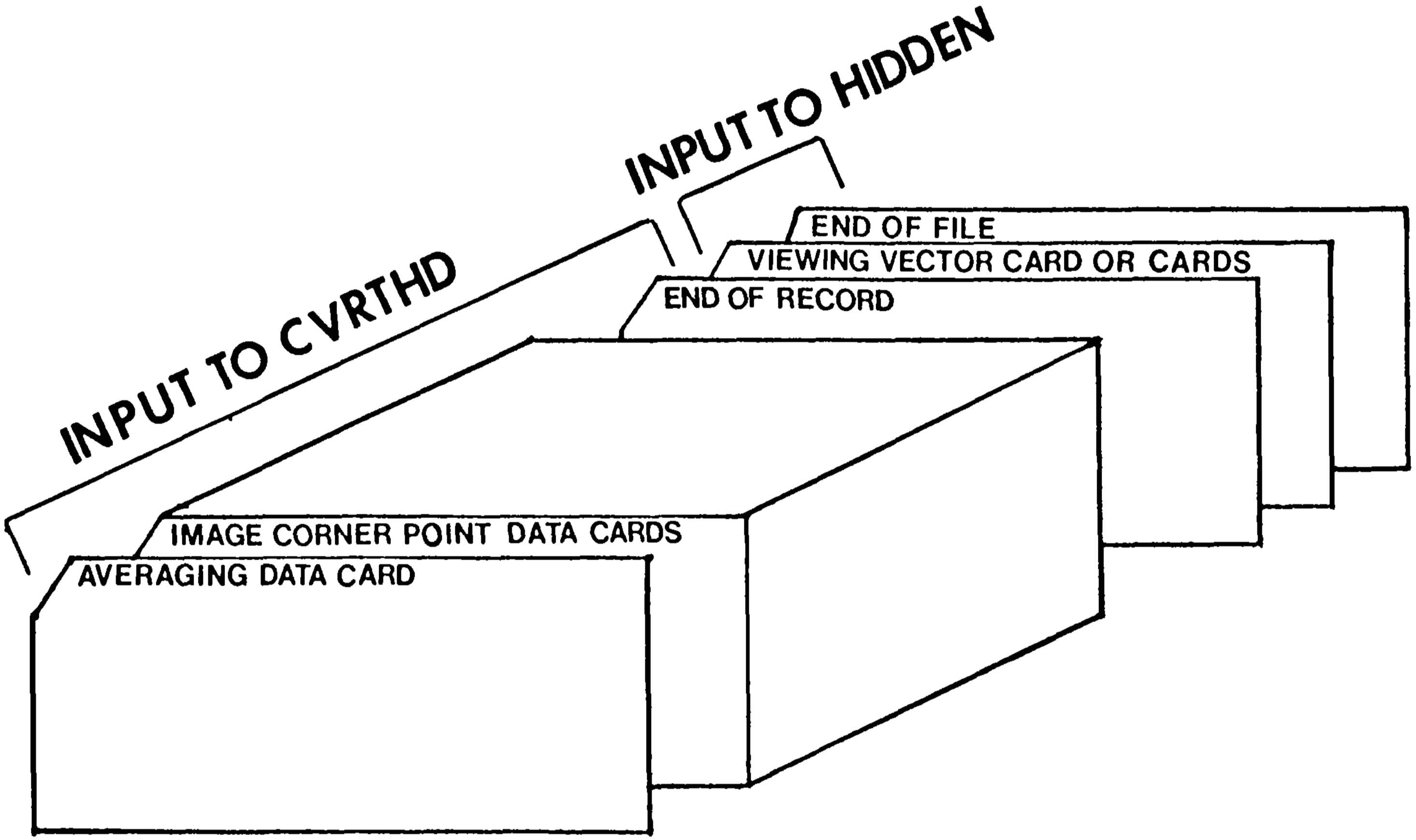


FIGURE 27. INPUT DATA DECK TO CONVERT IMAGE DATA TO HIDDEN LINE PICTURE

Program PLOTHD - Picture Drawer

Program PLOTHD reads the scale information and line segment corner points on file HIDOUT and prepares the X-Y plotter file using the Aerophysics Research Corporation Plotting Software Package (PLOTTR), reference 7, or Langley Plotting Software. A flow chart of PLOTHD is given in Figure 28.

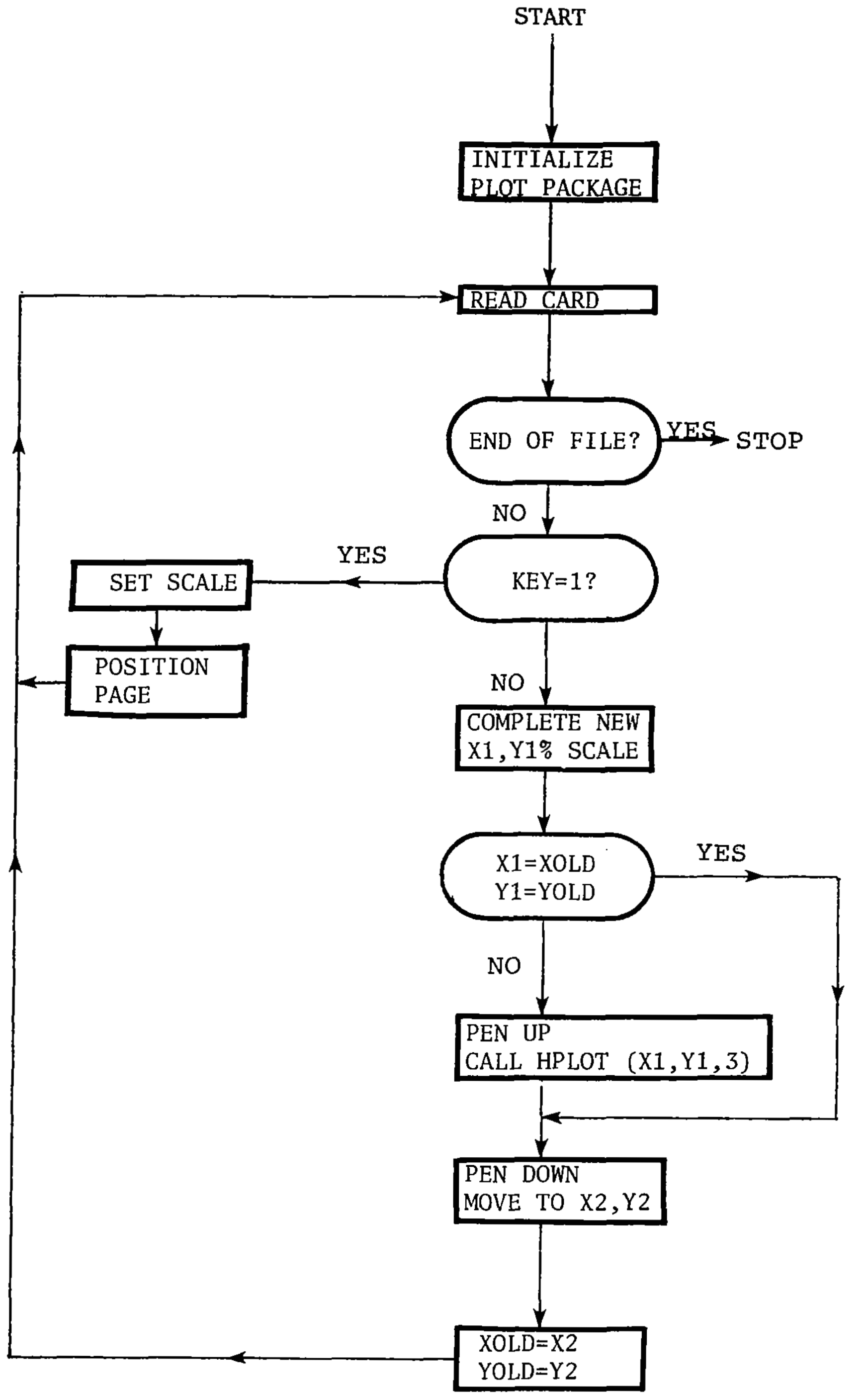


FIGURE 28. FLOW CHART FOR PLOTHD

APPLICATION TO THE FDL-6 CONFIGURATION - AN ILLUSTRATIVE PICTURE SEQUENCE OF HIDDEN LINE PICTURES

As an illustrative example of hidden line drawing ability the FDL-6 vehicle data used in Part I was supplied to PCSYS. The data was converted to HIDDEN format by CVRTHD. Program HIDDEN then prepared the hidden line data which was drawn on the Houston plotting device connected to a CDC 7600. The PLOTTR software was used to draw the resulting picture sequence for a series of viewing angles around the FDL-6 body. The fidelity of the resulting pictures is apparent in Figure 29. These pictures should be compared to those presented in Part I, Figures 10(a) and 10(b), which were produced by the more rapid running program IMAGE. In particular, attention is drawn to the upper three-quarters rear view of Figure 10(b) where the vehicle lower surface and interior is lost in the IMAGE picture. In the PCSYS pictures it is possible to see inside the vehicle from a reward direction since base panels were omitted.

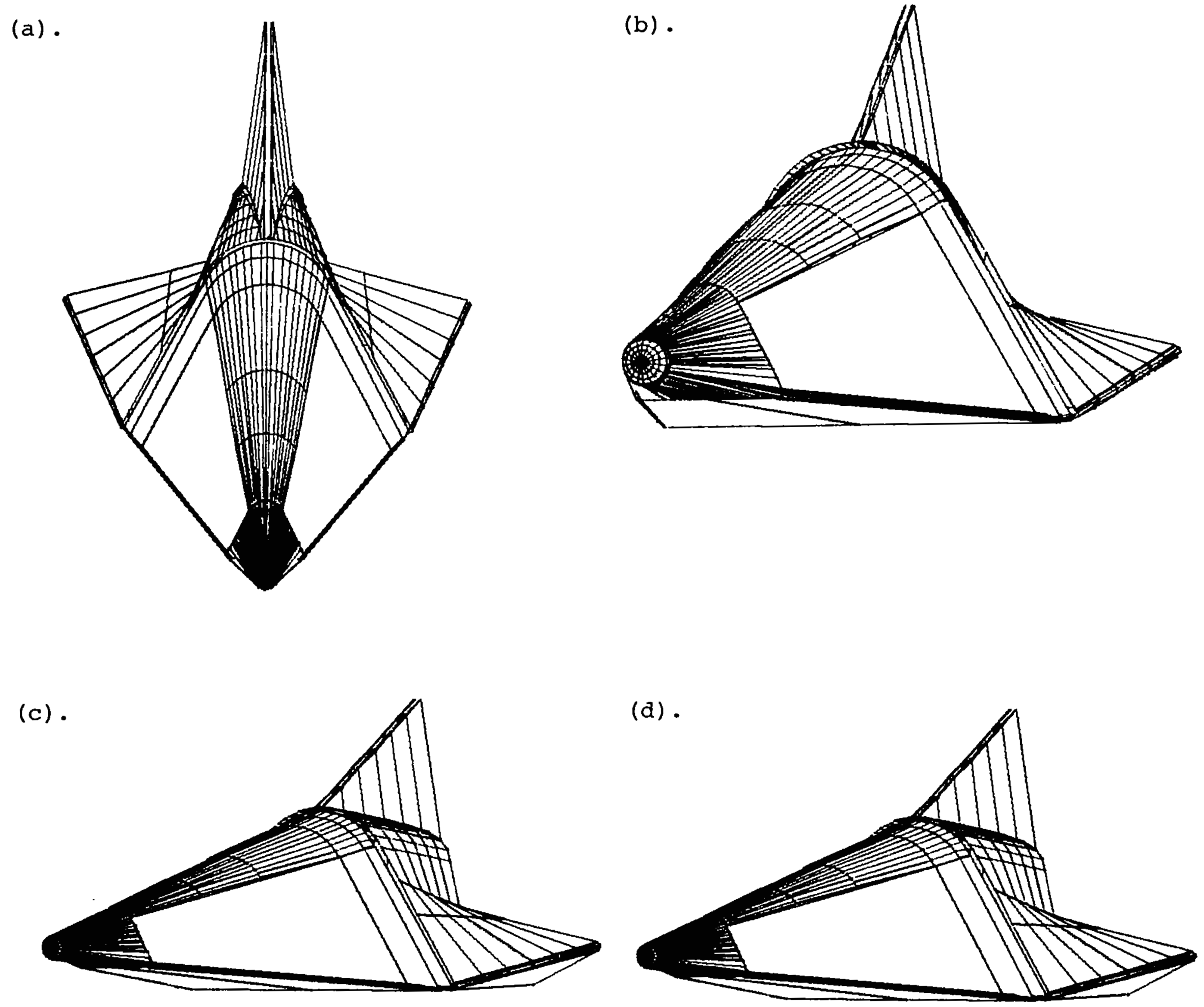
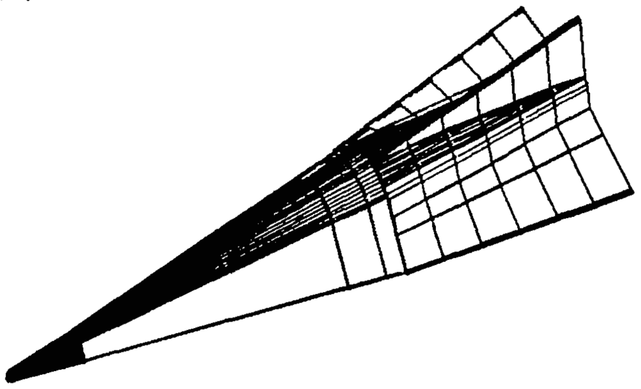
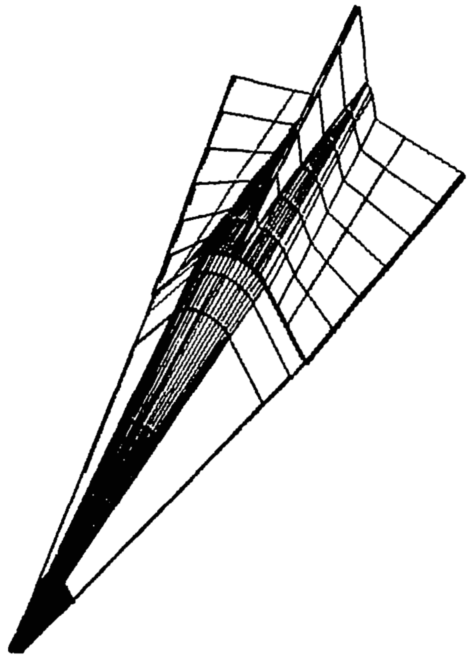


FIGURE 29. FDL-6 CONFIGURATION FROM PROGRAM HIDDEN

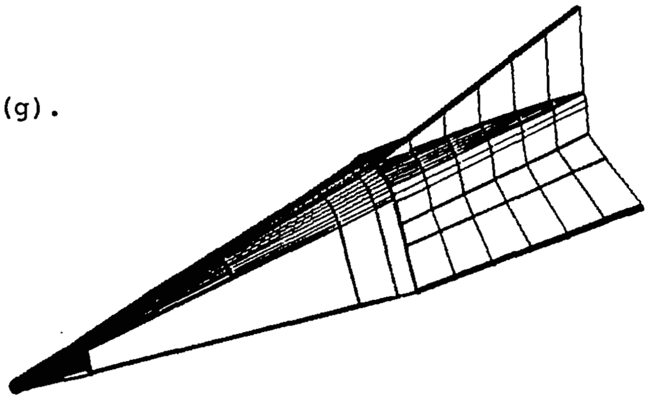
(e).



(f).



(g).



(h).

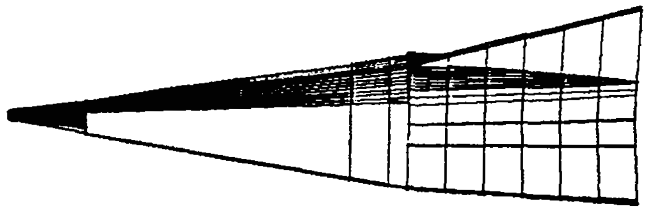
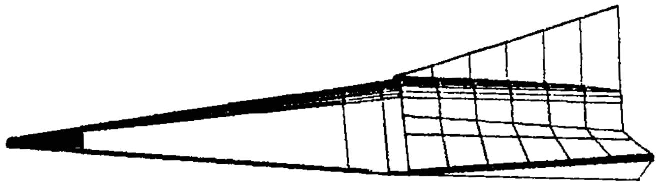
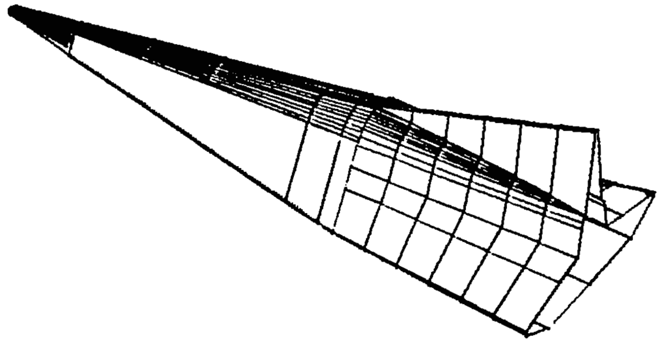


FIGURE 29(e-h). FDL-6 CONFIGURATION

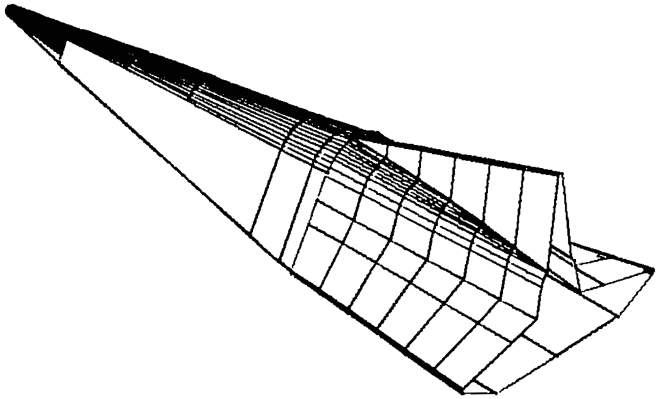
(i).



(j).



(k).



(l).

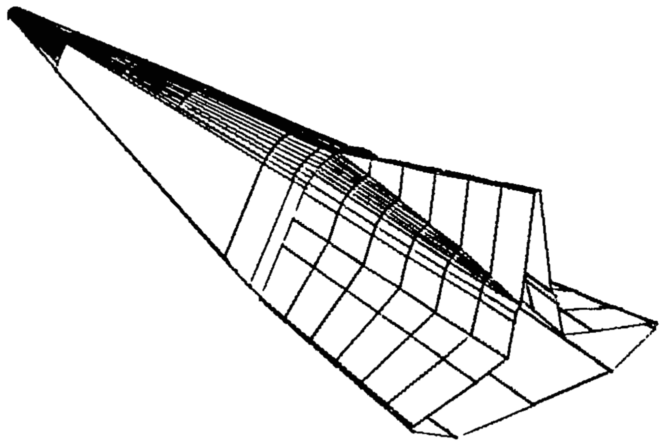
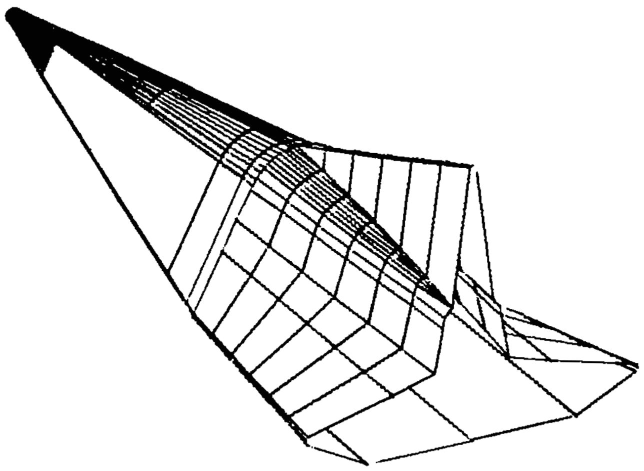
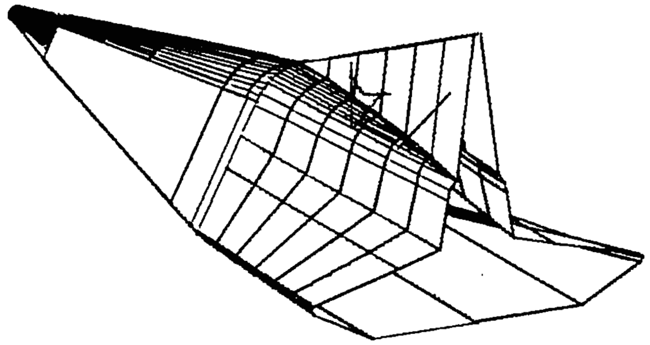


FIGURE 29(i-l). FDL-6 CONFIGURATION

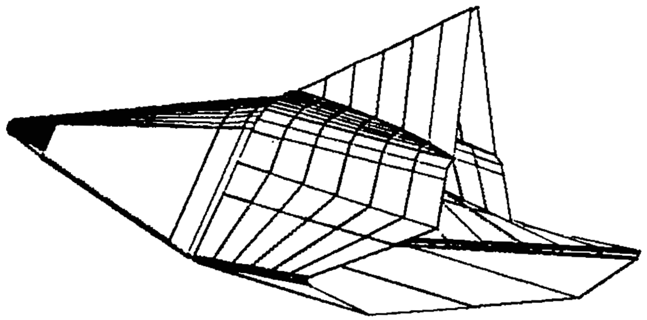
(m).



(n).



(o).



(p).

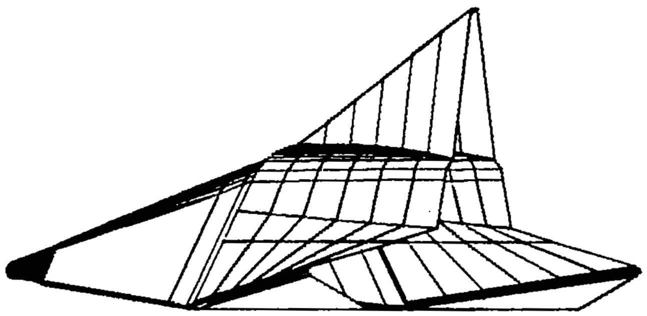


FIGURE 29(m-p). FDL-6 CONFIGURATION

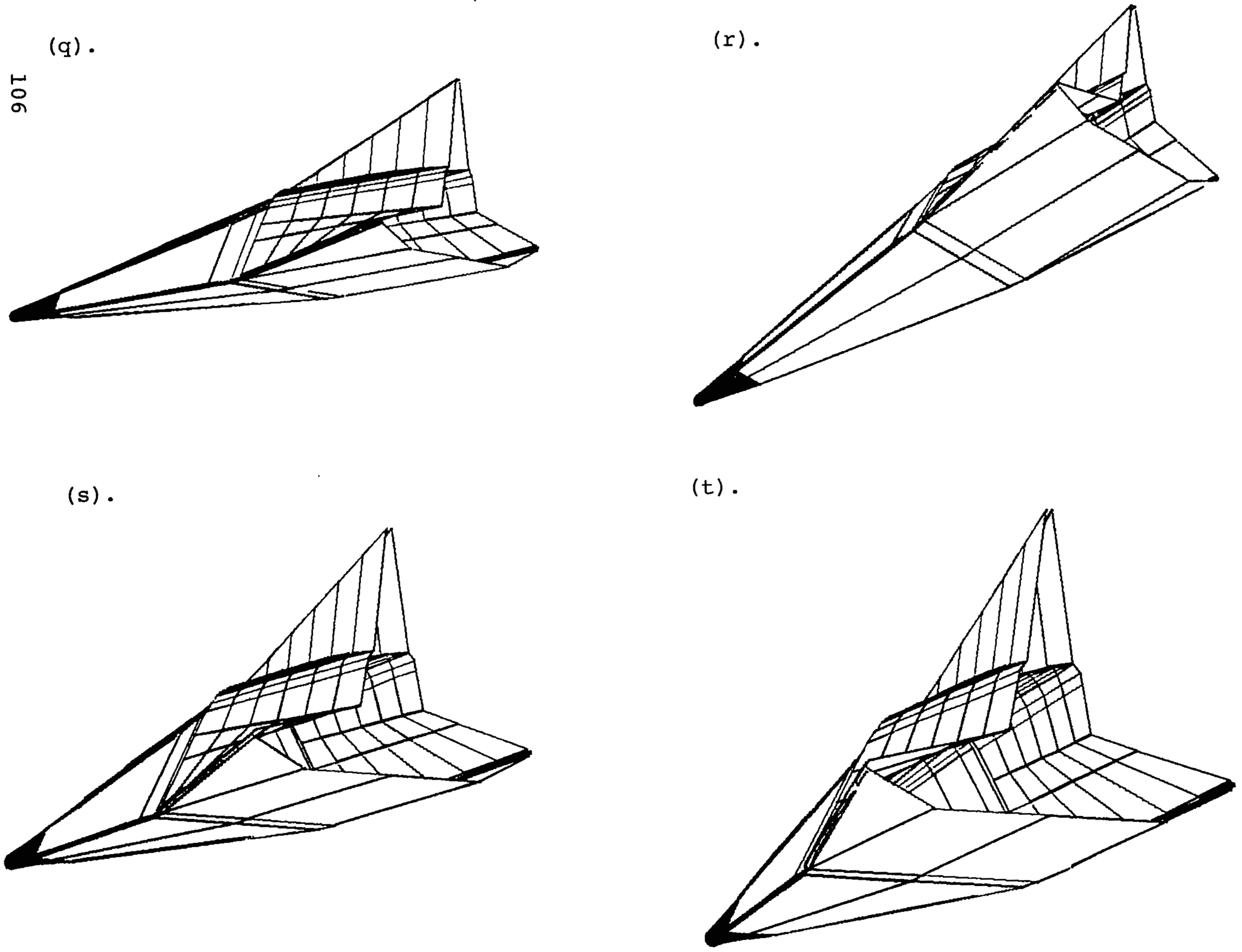
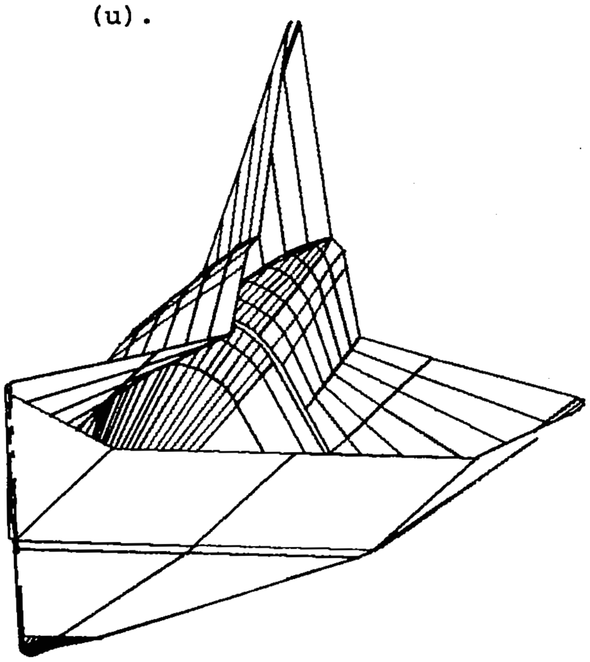
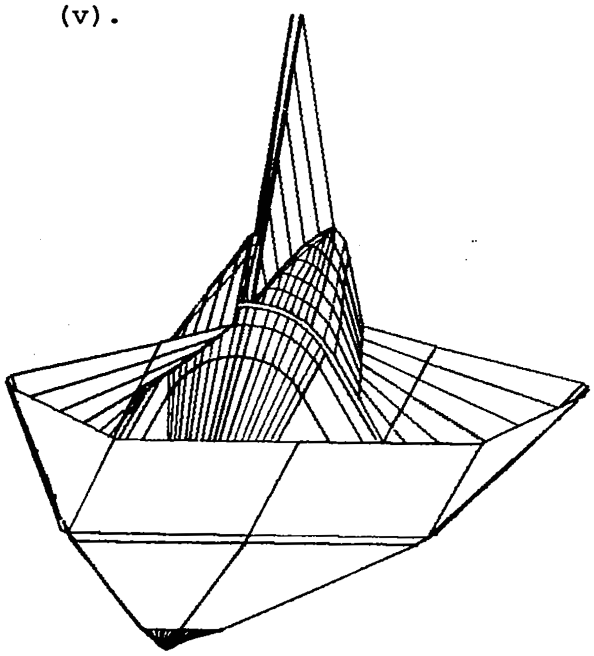


FIGURE 29(q-t). FDL-6 CONFIGURATION

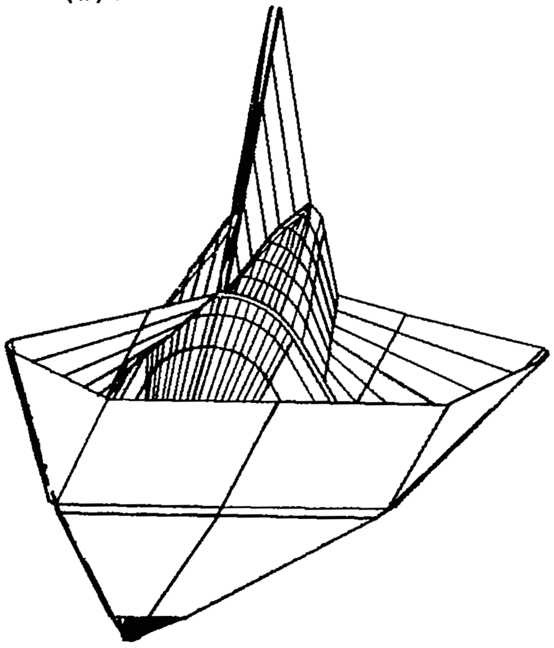
(u).



(v).



(w).



(x).

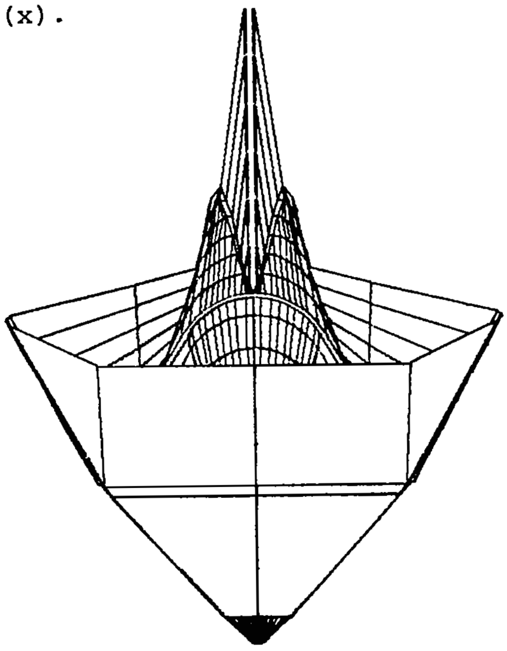


FIGURE 29(u-x). FDL-6 CONFIGURATION

APPLICATION TO AN ADVANCED TRANSPORTATION SYSTEM CONFIGURATION

A final demonstration applies to IMSYS to an Advanced Transportation System configuration derived from the ODIN system by NASA personnel at Langley Research Center. The pictures obtained are presented in Figure 30. These pictures are notable for the ability of the codes to draw a complex picture illustrating four aileron positions and again to examine the vehicle interior from rearward positions. Once again these pictures were drawn from data originally supplied to the IMAGE code. Typical pictures provided by IMAGE from the same data are presented in Figure 31.

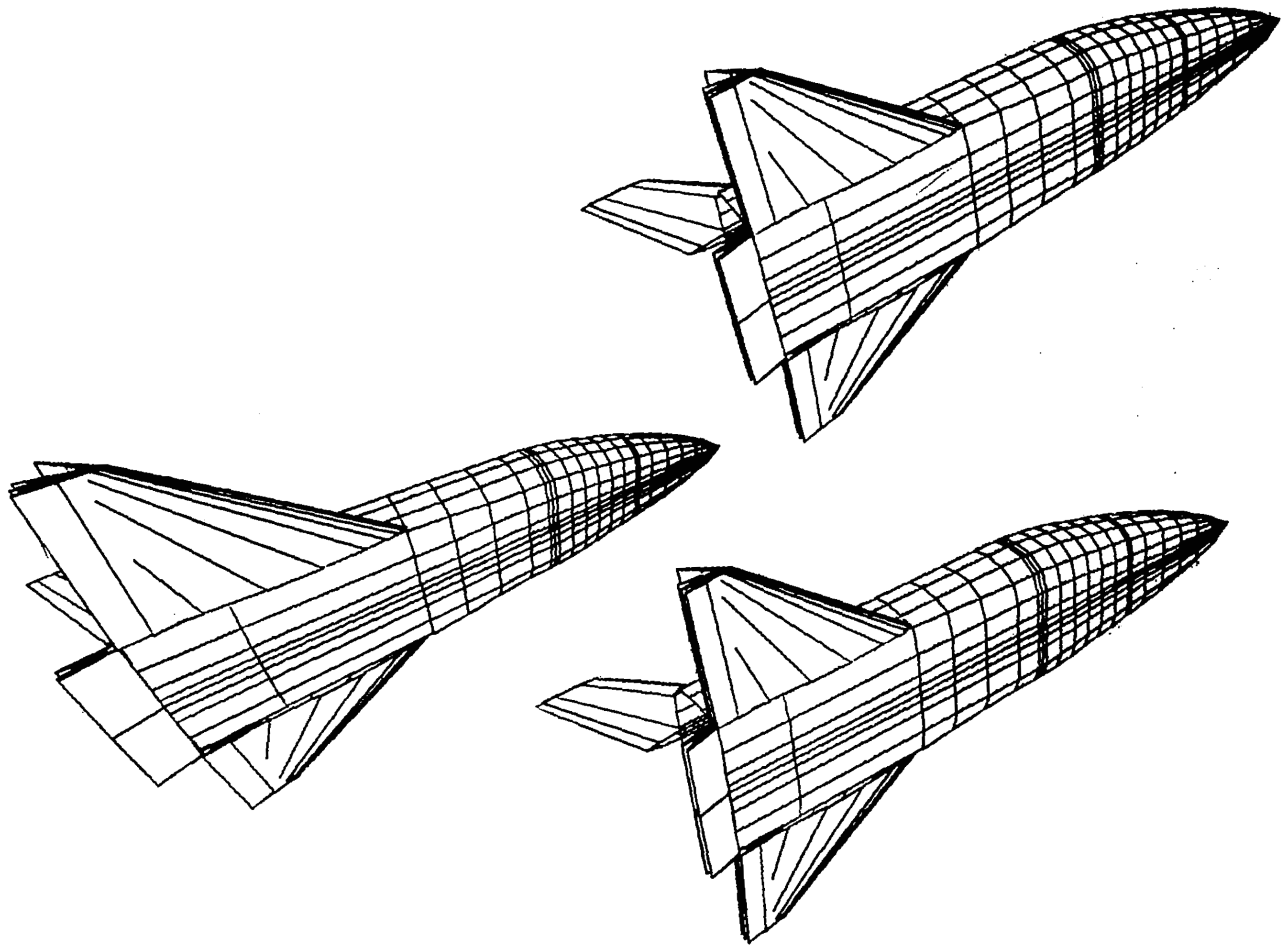


FIGURE 30(a). SOME TYPICAL ADVANCED TRANSPORTATION VEHICLE VIEWS

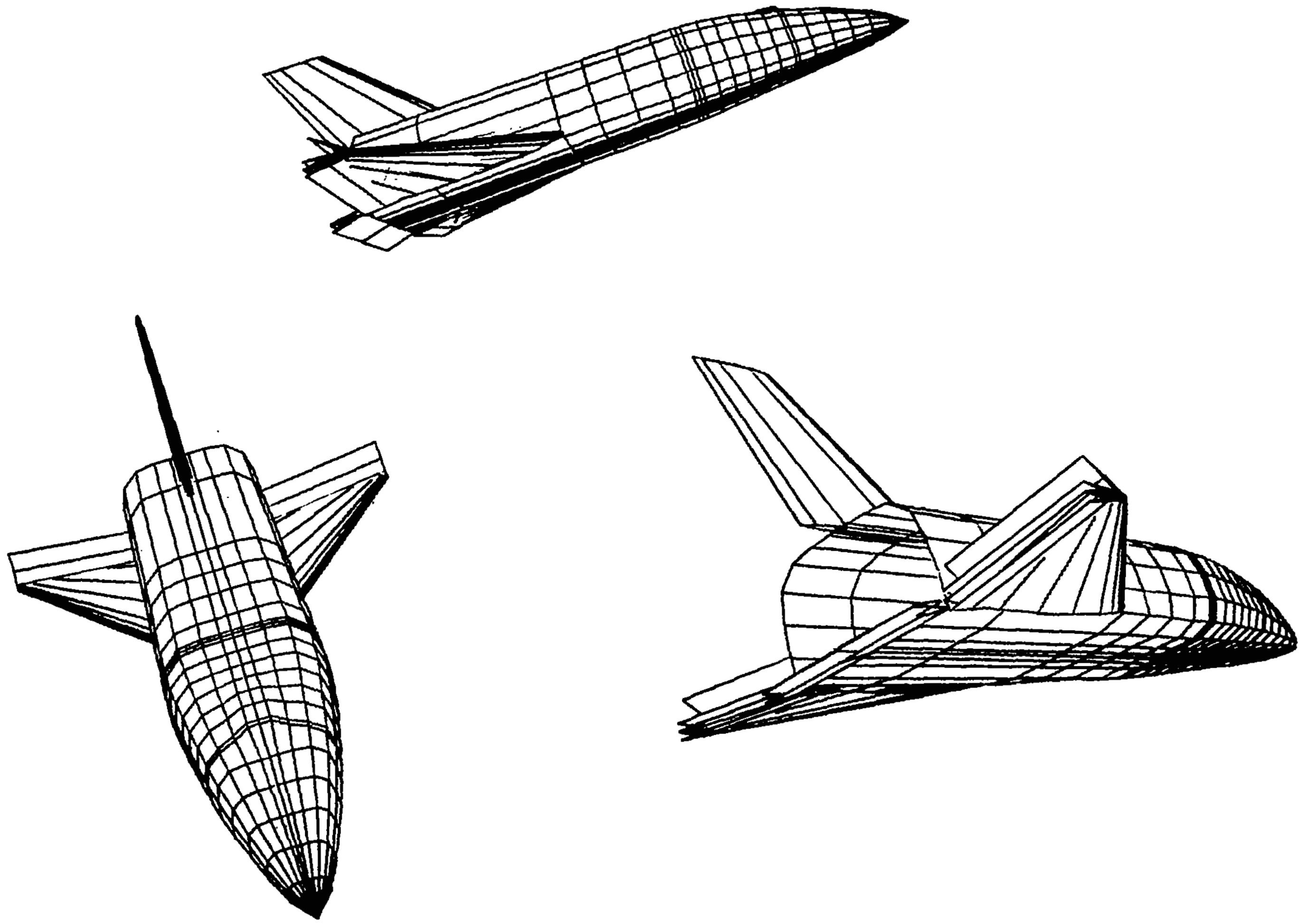


FIGURE 30(b). SOME TYPICAL ADVANCED TRANSPORTATION VEHICLE VIEWS FROM PROGRAM HIDDEN

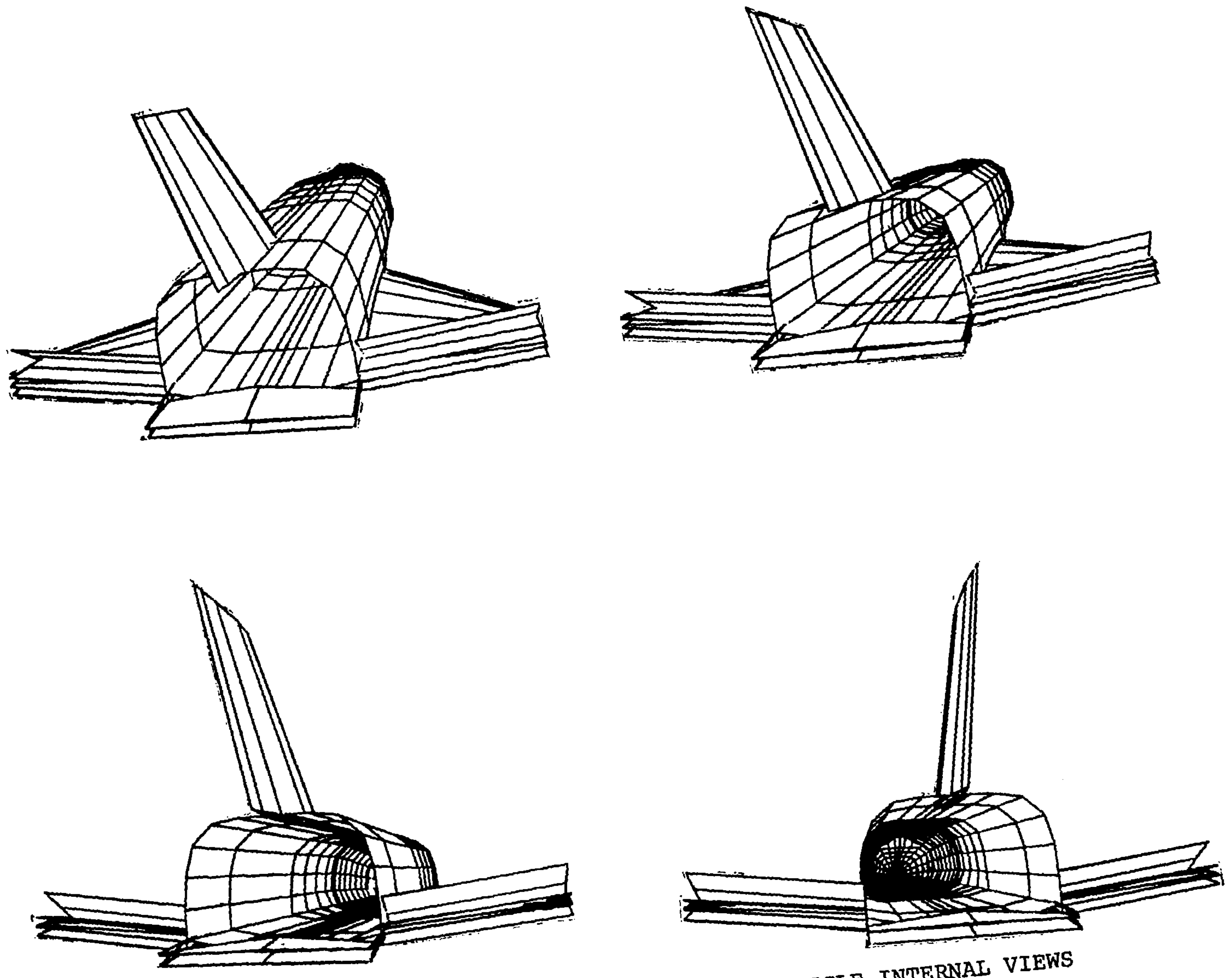


FIGURE 30(c). SOME ADVANCED TRANSPORTATION VEHICLE INTERNAL VIEWS

```
C O I N / I M A G E     PICTURE DRAWING PROGRAM
     NAVY ADVANCED TRANSPORTATION SYSTEM  BODY LENGTH=192.680

                                  VEHICLE CHARACTERISTICS

GEOMETRY
   BODY LENGTH.                                   192.7
   BODY VOLUME.                                   106360.7
   BASE AREA.                                     1252.2
   WING AREA.                                     6750.00
WEIGHTS
   DRY.                                           296012.8
   LANDING.                                       382137.3
   ENTRY.                                         392137.3
   PROPELLANT.                                    962667.53 /1806044.2
   GROSS.                                         3171923.9
PROPULSION
   NUMBER OF ENGINES                              2.640      /3.260
   VAC THRUST.                                    735300.0   /807328.0
   VAC ISP.                                       345.6      /466.5
   T/W                                            1.265      /1.191
PERFORMANCE
   IDEAL VELOCITY.                                23551.5
   STRUCTURAL WT IMPROVEMENT.PERCENT              25.0
   PAYLOAD.                                       65000.0
```

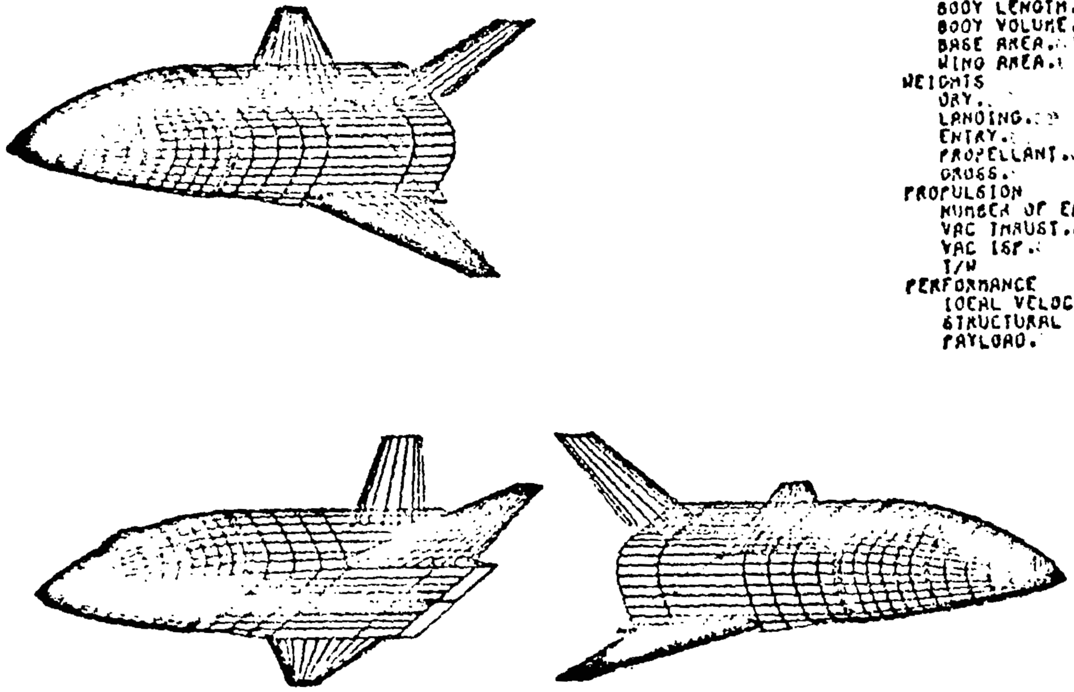


FIGURE 31. CONFIGURATION FROM IMAGE

CONCLUSION

An improved picture drawing code (PCSYS) for the ODIN Optimal Design Integration System has been developed. The picture drawing code retains the input format of the original ODIN picture drawing program IMAGE which, in general, has been well accepted by users of the ODIN system. The improved picture drawing code has the ability to produce plots with no hidden line removal, with a rapid approximate hidden line removal algorithm based on inclination of the outward normal, or finally with an accurate but time consuming hidden line removal algorithm.

This combination of picture drawing options enhances the graphics capability of the ODIN system while maintaining ease of user operation. The study has therefore contributed to the long range goal of improving and expanding the ODIN system technology modules.

REFERENCES

1. Gentry, Arvel: Hypersonic Arbitrary-Body Aerodynamic Computer Program. Vol. I, Douglas Report DAC 56080, June 1967.

2. Hague, D. S. and Glatt, C. R.: Optimal Design Integration of Military Flight Vehicles--ODIN/MFV. AFFDL-TR-72-132, December 1972.

3. Glatt, C. R. and Hague, D. S.: ODIN: Optimal Design Integration System. NASA CR-2492, February 1975.

4. Hague, D. S. and Glatt, C. R.: Optimal Design Integration of Military Flight Vehicles. Ch. 3.2, Program IMAGE: A Computer Code for Display of Three-Dimensional Objects, AFFDL-TR-72-132, December 1972.

5. Glatt, C. R.: IMAGE: A Computer Code for Generating Picture-Like Images of Aerospace Vehicles. NASA CR-2340, September 1974.

6. Hague, D. S. and Glatt, C. R.: Optimal Design Integration of Military Flight Vehicles. Ch. 3.1, Program PANEL: A Computer Code for Generating a Panelled Aerospace Vehicle Surface Definition, AFFDL-TR-72-132, December 1972.

7. Hague, D. S. and Glatt, C. R.: Optimal Design Integration of Military Flight Vehicles--ODIN/MFV, AFFDL-TR-72-132, Ch. 12-Program Plotter - Independent Plot Program, 1972

1. Report No. NASA CR-2912	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle PCSYS: The Optimal Design Integration System Picture Drawing System with Hidden Line Algorithm Capability for Aerospace Vehicle Configurations		5. Report Date December 1977
		6. Performing Organization Code
7. Author(s) D. S. Hague and J. D. Vanderberg		8. Performing Organization Report No. ARC-TN-218
		10. Work Unit No.
9. Performing Organization Name and Address Aerophysics Research Corporation P. O. Box 187 Bellevue, Washington 98009		11. Contract or Grant No. NAS 1-12977
		13. Type of Report and Period Covered Contractor Report
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, D. C. 20546		14. Sponsoring Agency Code

15. Supplementary Notes
Langley Technical Monitor: Timothy R. Rau
Final Report

16. Abstract

The PCSYS programs use a vehicle geometric definition based upon quadrilateral surface elements to produce realistic pictures of an aerospace vehicle. It is anticipated that the PCSYS system will be an important component of the Optimal Design Integration (ODIN) system, used in vehicle preliminary design studies. The PCSYS programs can be used to visually check geometric data input, monitor geometric perturbations, and to visualize the complex spatial inter-relationships between the internal and external vehicle components. The pictures constructed with PCSYS can be annotated with test information, if desired.

PCSYS has two major component programs. The first program, IMAGE, draws a complex aerospace vehicle pictorial representation based on either an approximate but rapid hidden line algorithm or without any hidden line algorithm. The second program, HIDDEN, draws a vehicle representation using an accurate but time consuming hidden line algorithm. Program IMAGE is based on the widely distributed code of Gentry. HIDDEN is based on a code developed by researchers at Purdue University.

17. Key Words (Suggested by Author(s)) computer graphics		18. Distribution Statement unclassified unlimited Subject Category 61	
19. Security Classif. (of this report) unclassified	20. Security Classif. (of this page) unclassified	21. No. of Pages 118	22. Price* $5.50

* For sale by the National Technical Information Service, Springfield, Virginia 22161